Programmazione dello ZX Spectrum

Tim Hartnell

Prefazione a cura di
CLIVE SINCLAIR

GRUPPO EDITORIALE JACKSON

EDIZIONE ITALIANA

Programmazione dello ZX Spectrum

Tim Hartnell

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

Il Gruppo Editoriale Jackson ringrazia per il prezioso lavoro svolto nella stesura dell'edizione italiana la signora Francesca Di Fiore, e l'Ing. Roberto Pancaldi.

Stampato in Italia da:
S.p.A. Alberto Matarelli – Milano – Stabilimento Grafico

Fotocomposizione:
CorpoNove s.n.c. – Bergamo – via Borfuro 14/c – Tel. 22.33.65

SOMMARIO

PREFAZIONE

Un foglio di carta può contenere gli scarabocchi di uno stupido o un sonetto di Shakespeare, ma da solo non significa nulla. Allo stesso modo le possibilità di un computer, per quanto potente possa essere, dipendono solo dall'abilità del programmatore; ecco perchè l'arte della programmazione ha assunto una grande importanza. Da qui questo libro, che rappresenta un utile complemento al manuale fornito con lo Spectrum. Nessun manuale può dire tutto e i nostri possono dare soltanto le basi della programmazione. Questo libro, per mezzo di esempi e programmi, vi aiuta a raggiungere una abilità di programmazione che, senza il suo aiuto, potrebbe richiedere anni per essere acquisita. Sarà molto utile sia al lettore che ha appena iniziato a programmare, sia al programmatore più esperto che desidera aumentare la sua abilità.

CLIVE SINCLAIR

RINGRAZIAMENTI

Molte persone mi hanno aiutato a scrivere questo libro, e desidererei ringraziarli sinceramente per avermi prestato la loro esperienza. Questo mi ha permesso di rendere il libro molto più comprensibile di quello che sarebbe risultato se lo avessi scritto basandomi solo sulle mie risorse. Diversi libri, menzionati al termine dei capitoli, sono stati di grande aiuto. Essi sono citati come guida per ulteriori letture, nel caso desideriate approfondire il soggetto specifico del capitolo. Coloro che hanno contribuito a specifiche parti del libro sono: Mike Salem, amministratore della Hilderbay, una delle più importanti ditte che producono software commerciale per lo ZX mi ha aiutato nell'introduzione del capitolo sui programmi finanziari e nell'appendice sulle parole in gergo; Jeff Warren, un esperto insegnante e capo della CALPAC Computer Software una ditta che produce software educativo ha aiutato nel capitolo sull'educazione e ha fornito cinque programmi; Tim Rogers, uno studente di Richmond, che ha dimostrato il suo entusiasmo per i giochi per computer nel capitolo 'GIOCARE CON LO SPECTRUM'; ha scritto BOMBARDAMENTO, JACKMAN, METEORE e BREAKOUT; Jeremy Ruston, un altro studente, che ha tre libri al suo attivo (Pascal for human beings, The BBC micro revealed, e The book of listing, un libro della BBC Publications che ha scritto in collaborazione con me) ha contribuito al capitolo sulla grafica tridimensionale; il dottor Tim Langdell, un esperto programmatore di West Dulwich, che scrive regolarmente per le riviste Your Computer e ZX Computing e che attualmente conduce un progetto di ricerca alla Open University sugli elaboratori, ha scritto i capitoli sul suono e sul colore; James Walsh, uno studente di Loughton che al momento sta scrivendo due libri (uno sulla costruzione di un computer dal nulla, e una dettagliata guida sull'uso del linguaggio macchina dello Spectrum) e che risponde alle domande dei lettori sulla rivista Personal Computer World, ha corretto il software per la mia rivista ZX Computing e ha contribuito al capitolo sul codice macchina. Infine devo ringraziare Clive Sinclair non solo per avere inventato lo Spectrum e i suoi predecessori, senza i quali io e centinaia di migliaia di altri non saremmo mai potuti entrare nel mondo dei computer, ma anche per il suo incoraggiamento e la sua assistenza.

INTRODUZIONE

La compilazione del dizionario della Oxford University Press ha richiesto 70 anni, perchè i suoi autori volevano includere tutte le parole della lingua inglese, e tutti i significati di ogni parola. Studiando la struttura di questo libro, mi accorsi che nello scriverlo avrei impiegato anche io 70 anni, se avessi incluso tutte le possibili applicazioni del computer. Così seleziona quelli che mi sembravano gli elementi più importanti della programmazione sullo Spectrum, e le aree in cui dovreste trovare più interessante l'uso del computer, e fondai la base del libro su queste conclusioni.

Molti dei miei libri sui computer includono una frase del tipo "Questo é un manuale più che un libro", con la quale intendevo che il libro doveva essere visto come uno strumento per apprendere l'uso del computer piuttosto che un libro da leggere come un romanzo. La stessa avvertenza si applica a questo volume. Così si potrebbero trarre indubbi vantaggi soltanto dalla lettura di esso, ma il vero valore del libro vi apparirà soltanto se lo leggerete con il vostro Spectrum a fianco, introducendo tutti i programmi mano a mano che li incontrate nel testo.

Non c'è alcun motivo per leggere il libro dall'inizio alla fine nell'ordine in cui è stato scritto. Ci possono essere cose che già conoscete, e che per ora non vi interessano. Saltate questi capitoli la prima volta che li incontrate, potrete eventualmente ritornarci in seguito.

Qualunque cosa decidiate, non considerate questo volume come un vero e proprio libro di testo. Ho sempre pensato che i libri di testo siano pervasi da una certa aria di superiorità che rende il loro contenuto noioso, e questa è l'ultima cosa che volevo ottenere con questo libro. Esso è una guida attraverso le aree dell'uso del computer dove possono anche accadere avventure.

Se lo guarderete in questo modo il libro vi aiuterà a trarre grandi soddisfazioni dal vostro Spectrum.

Inizieremo esaminando i concetti fondamentali della programmazione, poi aggiungeremo suono e colore ai vostri programmi, usando i comandi appositi per far produrre allo Spectrum effetti più interessanti di quanto mai abbiate creduto possibile.

Successivamente scopriremo come usare lo Spectrum negli affari e nell'educazione con un buon numero di programmi ampliamente spiegati e ancora un grande capitolo sui giochi. Questo include programmi per diversi giochi, pronti per essere usati e dovrebbe anche darvi qualche idea per creare da soli i vostri giochi. In questo capitolo esamineremo anche lo sviluppo della grafica definita dall'utente.

Un uso più spettacolare dell'alta risoluzione grafica dello Spectrum é considerata nel capitolo successivo per la produzione di figure tridimensionali.

Infine esamineremo l'uso del codice macchina dello Spectrum; troverete anche diversi suggerimenti per migliorare la vostra capacità di programmatori e nell'appendice una breve storia dei computer, una spiegazione dei termini tecnici più usati e una scheda delle specifiche tecniche dello Spectrum.

Ma soprattutto, spero che siamo riusciti a produrre una guida utile e comprensibile per il vostro ZX Spectrum, che vi aiuterà a trarre il meglio dal vostro computer per il futuro.

Buona programmazione.

TIM HARTNELL

CAPITOLO 1

PROGRAMMARE IN BASIC

Il BASIC è il linguaggio di programmazione più conosciuto nel mondo, perchè è il più facile da imparare. I linguaggi sono divisi in livelli: un linguaggio ad alto livello é molto vicino all'Inglese, uno a basso livello è più vicino agli astrusi schemi di 1 e 0 che un calcolatore é in grado di capire.

Il BASIC é un linguaggio ad alto livello. Anche se non avete esperienza di programmazione sarete senz'altro lieti di imparare già un po' di BASIC.

Parole come PRINT (stampa), STOP e AND significano in BASIC esattamente quanto significano in Inglese. Cosi come IF (se) e THEN (quindi, allora) e OR (oppure). Penso che vi stiate già rendendo conto che la programmazione è più semplice di quanto avevate pensato.

Essenzialmente quando scrivete un programma date allo Spectrum una serie di istruzioni da seguire. Lo Spectrum, come tutti i calcolatori costruiti fino ad oggi ha delle straordinarie capacità di calcolo e decisionali, ma é assolutamente privo di immaginazione. Se dite ad un computer di fare qualcosa esso lo farà. Se omettete parte delle istruzioni cercherà di realizzare il resto senza rendersi conto che qualcosa è stato omesso.

Immaginate di avere un robot e che vogliate farvi preparare il bagno. GO TO BATH (vai alla vasca) potrebbe essere la prima istruzione del programma. IF BATH IS EMPITY THEN TURN ON TAP. IF BATH IS FULL THEN TURN TAP OFF. (Se la vasca é vuota allora apri il rubinetto. Se la vasca é piena allora chiudi il rubinetto).

Il robot entrerebbe allegramente nella stanza, immergerebbe la sua mano elettronica nella vasca e scoprirebbe che è vuota, quindi aprirebbe il rubinetto. Poi starebbe lì per sempre aspettando che il livello dell'acqua raggiunga un punto dove possa essere controllato, prima di richiudere il rubinetto. Ma poichè avete dimenticato di includere una istruzione del tipo: CONTROLLA SE IL TAPPO È NEL BUCO E SE NON C'È ALLORA METTI IL TAPPO NEL BUCO il robot non ci penserebbe. Il vostro Spectrum, seguendo le vostre istruzioni, procede esattamente in questo modo. E credeteci o no, con qualche riserva il vostro Spectrum potrebbe eseguire le istruzioni che avreste dato al robot... perché quelle istruzioni contengono un certo numero di parole del linguaggio BASIC.

Qui c'é un semplice programma per lo Spectrum che sono sicuro potrete capire anche senza alcuna cognizione del linguaggio dei computer:

```
LET A=20
LET B=A+A
IF B=40 THEN PRINT "B EQUIVALE A 40"
```

Questo é rigorosamente Inglese, ma é anche BASIC.

La prima linea del programma LET A=20 (let significa poni) é semplice, così come la seconda (LET B=A+A).

La terza assomiglia molto ad una delle prime istruzioni date al robot: "Se la vasca é piena allora chiudi il rubinetto"; se B=40 allora stampa "B EQUIVALE A 40". Potreste battere questo programma nello Spectrum ed esso farebbe immediatamente stampare al calcolatore "B EQUIVALE A 40".

L'unica cosa di cui queste tre linee hanno bisogno per diventare un programma é un numero che le identifichi.

L'esempio seguente é un programma che potrebbe girare sul vostro Spectrum:

```
10 LET A=20
20 LET B=A+A
30 IF B=40 THEN PRINT "B EQUIVALE A 40"
```

I numeri di linea possono essere un qualsiasi numero a vostra scelta (compreso tra 1 e 9999). Il calcolatore le ordina automaticamente. È preferibile numerare le linee di 10 in 10 in quanto questo procedimento lascia spazio tra le linee e ci permette in seguito di aggiungerne altre.

Come vedete avete già fatto un po' d'esperienza con le parole BASIC, LET, IF... THEN, e PRINT. I segni uguale e più li avete imparati già dall'aritmetica. In molti casi essi si comportano in BASIC esattamente come fanno in una comune somma.

Prima di accendere effettivamente lo Spectrum e di programmarlo, vorrei farvi conoscere un'altra parola BASIC il cui significato vi apparirà subito chiaro. In effetti le parole sono in realtà due: GO e TO (VAI e A).

In BASIC esse appaiono sempre insieme e sono disponibili su di un singolo tasto dello Spectrum (il tasto G). Vi ricorderete che abbiamo fatto iniziare al robot il programma 'Preparazione del bagno' dicendogli GO TO BATH.

Nel mondo della programmazione si dice GO TO numero di linea. Potremmo aggiungere una linea finale chiamata 40 alle tre già esistenti nel programma precedente, che dice:

```
40 GO TO 30
```

Questo significa che il computer eseguirebbe la linea 30 e quindi stamperebbe "B EQUIVALE A 40", poi procederebbe in sequenza fino alla linea 40 dove troverebbe l'istruzione GO TO 30.

Senza domandarsi perché gli avete detto di fare così, il calcolatore seguirebbe le istruzioni e ritornerebbe alla linea 30 dove troverebbe l'istruzione di stampare "B E-

QUIVALE A 40' che eseguirebbe per la seconda volta, procederebbe poi alla linea 40 dove sarebbe rimandato alla 30 e andrebbe avanti così fino a che lo schermo fosse pieno di "B EQUIVALE A 40".

Ma non possiamo andare oltre senza avere acceso lo Spectrum, perciò prendete il vostro computer e seguendo le istruzioni del manuale collegatelo all'alimentatore e al televisore.

La tastiera dello Spectrum sembra complessa la prima volta che l'affrontate, con tutte quelle parole come MERGE (compattare) e VERIFY (verificare) e quei termini matematici come SIN e COS che avevate sperato di avere lasciato per sempre a scuola. Non preoccupatevi! Una volta conosciuta la tastiera essa vi apparirà semplicissima da controllare e come diceva lo slogan per i predecessori dello Spectrum: «In pochi giorni comincerete a parlargli come ad un vecchio amico".

LA TASTIERA

Due delle parti più importanti della tastiera sono i tasti di shift. Sono nella fila in basso: il CAPS SHIFT (in bianco) in basso nell'angolo a sinistra e il SYMBOL SHIFT (in rosso) sempre in basso, secondo da destra. Questi tasti determinano quello che otterrete quando premerete gli altri tasti, con il loro complesso insieme di parole e simboli.

Assumiamo da ora che il vostro computer sia acceso: premete uno qualsiasi dei tasti bianchi alfabetici e vedrete apparire la parola scritta sul tasto (esempio LOAD, LIST o PRINT).

Questa è una parola chiave. Lo Spectrum per risparmiare memoria e rendere più semplice la programmazione utilizza il sistema delle parole chiave (keyword).

Semplicemente premendo un tasto si stampa l'intera parola relativa al tasto. Nella maggior parte degli altri computer dovete battere una parola come LIST, DIM o FOR per intero.

Tenendo premuto il tasto CAPS SHIFT premete il tasto DELETE (in alto a destra) fino a che avrete cancellato tutto quello che avete scritto.

Ora premete il tasto P ed apparirà la parola PRINT. Premete qualche tasto numerico in modo da avere in fondo allo schermo qualcosa come PRINT 62735. Premete ora il tasto chiamato ENTER (a destra nella penultima fila). Scomparirà la scritta in basso e sullo schermo ricomparirà in alto il numero che avete richiesto. Il computer ha ubbidito alla vostra istruzione di stampare un numero. Generalmente lo Spectrum aspetterà fino a che non viene premuto ENTER prima di fare realmente qualcosa.

Introdurremo ora un semplice programma per mostrare l'uso delle parole chiave. Tenendo premuto il CAPS SHIFT e premendo il tasto 2 (CAPS LOCK) per ottenere le lettere maiuscole, introducete il seguente programma:

10 INPUT (basta premere il tasto I) A (adesso premete ENTER per indicare che avete terminato la linea di programma).
20 PRINT A (premete ENTER).
30 GO TO 10 (premete ENTER).

Notate che la parola PRINT viene dal tasto P e GO TO dal tasto G. Gli spazi nel listato del programma sono aggiunti automaticamente. Per fare funzionare il programma appena introdotto premete il tasto R, che farà apparire la parola RUN.

Premete quindi ENTER.

Il cursore lampeggiante apparirà in fondo allo schermo, mostrando che il computer è in attesa di scrivere il numero.

Introducete qualsiasi numero, quindi premete ENTER: vedrete apparire il vostro numero in cima allo schermo. Sapete dalla discussione precedente su GO TO BATH, che l'ultima linea del programma (linea 30) rimanda l'esecuzione alla linea 10. In questo modo il computer continuerà ad eseguire lo stesso programma fino a che non lo fermerete. Potete fermare questo programma introducendo qualsiasi lettera eccetto la A, quando lo Spectrum è in attesa di un numero. Il computer si fermerà stampando un messaggio di errore in fondo allo schermo.

Una volta fermato il programma premete il tasto A, premete ENTER e lo schermo diventerà scuro e successivamente chiaro e in fondo allo schermo apparirà l'avviso di Copyright. La parola chiave ottenibile dal tasto A é NEW che cancella tutto quello che risiede nella memoria del computer, quindi dovete porre molta attenzione nell'usarla.

Così avete imparato che le parole chiave bianche si ottengono premendo il tasto richiesto dopo aver introdotto un numero di linea.

Guardiamo ora le parole in rosso sui tasti. Teniamo premuto il tasto SYMBOL SHIFT e premiamo poi il tasto Y; vedrete comparire la parola AND. Sempre tenendo premuto il SYMBOL SHIFT, premete la G e apparirà THEN. Quindi le parole rosse sui tasti e i simboli come + si ottengono mantenendo premuto il SYMBOL SHIFT e premendo il tasto relativo.

Otterrete le parole in verde sopra i tasti premendo assieme i due tasti di shift, rilasciandoli e premendo quindi il tasto relativo. Provate: premete CAPS SHIFT e SYMBOL SHIFT assieme, lasciateli, quindi premete il tasto A: dovrebbe apparire la parola READ.

È un po' più difficile ottenere le parole in rosso sotto i tasti. Premete i tasti di shift assieme, quindi lasciate CAPS SHIFT, ma tenete sempre premuto SYMBOL SHIFT e premete il tasto X: dovrebbe apparire la parola INK.

Dovrete impratichirvi un po' per essere sicuri di ottenere la parola INK (o BEEP, PAPER, FLASH o BRIGHT) ogni volta.

Leggete ancora questa parte per intero, esercitandovi finché sarete sicuri di aver capito. Non preoccupatevi anche se vi sembrerà di impiegare molto tempo. Sarete piacevolmente colpiti di quanto rapidamente imparerete il sistema di funzionamento della tastiera. Al termine di questo capitolo introduttivo avremo un programma compilato apposta per insegnarvi dove sono i tasti alfabetici sullo Spectrum e per avviarvi alla "dattilografia", ma prima voglio presentarvi il primo vero programma di questo libro.

Potrebbe occorrervi un po' di tempo per batterlo, ma vi invito a persistere.

Una volta che avete introdotto tutto il programma premete RUN e giocate contro

lo Spectrum. Il programma é il gioco SASSO, FORBICI E CARTA, una versione computerizzata del noto gioco in cui due contendenti nascondono le mani dietro la schiena e assieme si mostrano reciprocamente la mano. Il pugno chiuso simboleggia il SASSO; l'indice e il medio distesi le FORBICI, mentre la mano aperta rappresenta il foglio di CARTA.

Il sasso vince sulle forbici (perchè può spuntarle), le forbici battono la carta (perchè possono tagliarla) e la carta sul sasso (perchè può avvolgerlo).

Introducete la vostra scelta fra sasso, forbici e carta tramite i numeri 1, 2 o 3. La scelta che essi rappresentano appare in cima allo schermo.

Prima di introdurre il programma i seguenti commenti potrebbero servirvi.

Potete ottenere il piccolo segno (c) di Copyright alla linea 20 premendo i due tasti di shift contemporaneamente e successivamente (tenendo premuto solo SYMBOL SHIFT) premendo il tasto P.

Il segno uguale (usato la prima volta alla linea 30) viene dal tasto L: lo ottenete tenendo premuto SYMBOL SHIFT e quindi il tasto L. Ricordatevi di premere ENTER dopo aver scritto ogni linea di programma, in modo da muovere la linea dal fondo alla cima dello schermo. Se la linea non si muove, un punto di domanda lampeggiante apparirà da qualche parte sulla linea: è un segnale del computer per avvertirvi che avete commesso un errore di sintassi.

Fate bene attenzione a dove sta lampeggiando il punto di domanda, in modo da scoprire il vostro errore. I due punti (:) dopo la parola CLS (Clear the Screen = cancellazione dello schermo) alla linea 60 si trovano sul tasto Z. Tenete premuto SYMBOL SHIFT e premete Z per ottenerli.

La linea 70 ha un apostrofo (') dopo la parola PRINT e prima delle virgolette ("). Troverete l'apostrofo sul tasto 7, e lo otterrete tenendo premuto il SYMBOL SHIFT e premendo il 7. Otterrete le virgolette dal tasto P.

Avrete bisogno di esercitarvi un poco per introdurre la linea 170 nel vostro calcolatore. Dopo 170 LET C=, avete bisogno della parola INT che non va battuta lettera per lettera, ma è ottenibile dal tasto R. Premete entrambi i tasti di SHIFT, rilasciateli, poi premete R e apparirà INT sullo schermo.

La parentesi aperta è sul tasto 8 (si ottiene tenendo premuto SHIFT mentre si preme il tasto 8) e la parola RND (generazione di un numero casuale) é ottenibile dal tasto T: premere entrambi gli SHIFT, lasciarli e poi premere T. L'asterisco (*) che in BASIC é il segno di moltiplicazione si ottiene dal tasto B: tenete premuto SYMBOL SHIFT mentre premete B.

Non provate a scrivere lettera per lettera parola come: AND, OR, THEN E TO, ma cercate di trovarle sui tasti; queste parole si ottengono tenendo premuto il tasto SYMBOL SHIFT, e premendo il tasto relativo. Ora caricate il programma e ritornate al libro.

```
10 REM SASSO, FORBICI, CARTA
20 REM (c) HARTNELL, 1982
30 LET COMP=0
```

```
  40 LET HUM=0
  50 FOR A=1 TO 10
  60 CLS : PRINT "GARA NUMERO "
;A
  70 PRINT '"1 - SASSO"
  80 PRINT "2 - FORBICI"
  90 PRINT "3 - CARTA"
 100 PRINT '"INTRODUCI 1, 2 O 3"
 110 INPUT B
 120 PRINT '"HAI SCELTO ";
 130 IF B=1 THEN PRINT "SASSO"
 140 IF B=2 THEN PRINT "FORBICI"
 150 IF B=3 THEN PRINT "CARTA"
 160 PAUSE 50
 170 LET C=INT (RND*3)+1
 180 PRINT '"IO HO SCELTO ";
 190 IF C=1 THEN PRINT "SASSO"
 200 IF C=2 THEN PRINT "FORBICI"
 210 IF C=3 THEN PRINT "CARTA"
 220 LET D=260
 230 IF B=C THEN PRINT '"E' UN PA
REGGIO!": GO TO D
 240 IF C=1 AND B=2 OR C=2
AND B=3 OR C=3 AND B=1 THEN
PRINT '"HO VINTO!": LET COMP=COMP
+1: GO TO D
 250 IF B=1 AND C=2 OR B=2 AND C=
3 OR B=3 AND C=1 THEN PRINT '"HAI
VINTO!": LET HUM=HUM+1
 260 PRINT '"PUNTEGGIO"''"IO> ";C
OMP;TAB 10;"TU> ";HUM
 270 PAUSE 100
 280 NEXT A
 290 IF HUM=COMP THEN PRINT ''"LA
GARA E' FINITA IN PAREGGIO!"
 300 IF HUM<COMP THEN PRINT ''"HO
VINTO QUESTA PARTITA!"
 310 IF COMP<HUM THEN PRINT ''"HA
I VINTO QUESTA PARTITA!"
```

Seguirò il programma passo per passo per provare a spiegare ciò che succede in ogni linea.

10 Questa linea inizia con REM che significa REMark (Commento). Le frasi REM sono incluse nei programmi a beneficio di chi legge il listato del programma; il computer ignora tutto quello che appare in una lista dopo la parola REM.

20 Questa é un'altra frase REM, anche questa ignorata dal computer.

30 LET significa la stessa cosa sia in BASIC che in Inglese. (Porre, dare un valore). La parola COMP non è una frase BASIC ma il nome di una variabile che può essere qualsiasi combinazione di lettere e numeri (la prima deve essere una lettera) a cui è stato assegnato un valore numerico.
COMP tiene il punteggio del computer ed è posto uguale a zero all'inizio del gioco.

40 Questa linea ha la stessa funzione della linea precedente, per la variabile HUM (punteggio del giocatore).

50 FOR... Questo è l'inizio di quello che é chiamato un ciclo di FOR/NEXT. Il computer generalmente compie questo ciclo per un numero di volte pari al numero posto alla fine della frase FOR. Se guardate la linea 280 (NEXT A) troverete la fine del ciclo. In questo programma il computer esegue il tratto tra la linea 50 e la 280 per 10 volte, portando a termine tutte le istruzioni all'interno del ciclo.

60 CLS, come detto prima, cancella lo schermo.
Il segno due punti (:) vi permette di aggiungere una seconda frase alla linea. PRINT è nel nostro caso la seconda frase. La sua funzione é appunto di stampare le parole comprese fra le virgolette. Oltre a ciò il computer stampa anche il VALORE di A.
Nella prima esecuzione del ciclo FOR/NEXT, A è uguale a 1; la seconda volta A è uguale a 2 e così via sino a che A é uguale a 10 nell'ultima esecuzione. Così PRINT "GARA NUMERO"; A produce GARA NUMERO 4, o qualsiasi altro numero che sia uguale ad A in quel determinato ciclo.

70-90 Queste 3 linee stampano i numeri da 1 a 3 e la parola che rappresentano nel gioco, ad esempio SASSO. Notate l'apostrofo (dal tasto 7) prima dell'apertura delle virgolette alla linea 70: così facendo quello che segue è stampato una linea più in basso, di modo che vi sia una linea vuota tra le parola GARA NUMERO 4 e 1 - SASSO.

100 L'apostrofo è usato ancora per mettere un'altra linea vuota prima di stampare: INTRODUCI 1, 2 0 3.

110 Il comando INPUT aspetta sino a che un numero è introdotto dall'utente. In questo caso il numero che voi introducete è assegnato alla variabile B.

120 Questa linea stampa HAI SCELTO.

130-150 Queste tre linee interpretano il numero che avete introdotto (1, 2 o 3 assegnato a B) e decidono a quale parola corrisponde (SASSO, FORBICI o CARTA) e di conseguenza cosa stampare sullo schermo.

160 PAUSE Ciò permette un leggero ritardo per far sembrare che il computer stia pensando.
Il numero dopo la parola PAUSE è espresso in cinquantesimi di secondo (in sessantesimi negli Stati Uniti), in questo caso PAUSE 0 compierà un ritardo indeterminato; questo stato di attesa avrà fine quando schiaccerete un tasto qualsiasi.

170 Questa è una linea molto interessante nella quale il computer genera un numero casuale. I numeri casuali sono molto usati nei programmi dei giochi. La funzione RND (entrambi i tasti SHIFT e quindi T) genera un numero compreso tra 0 e 1. Potete rendervi conto di ciò battendo PRINT RND e quindi ENTER: avrete stampato sul vostro schermo un numero compreso tra 0 e 1.
La linea 170 lo trasforma poi in un numero intero compreso tra 1 e 3. Se il 3 che segue l'asterisco dentro le parentesi fosse stato cambiato, per esempio con 10, C sarebbe stato posto uguale a un numero scelto casualmente dal calcolatore tra 1 e 10. Comunque in questo caso noi avevamo bisogno che il computer scegliesse 1, 2 o 3, così abbiamo moltiplicato RND per 3.

180 Questa linea inizia la frase che dice cosa ha scelto il computer.

190-210 Queste tre linee cambiano 1, 2 o 3, a cui C é stato posto uguale, in SASSO, FORBICI o CARTA.

220 La variabile D è posta uguale a 260: questo per far si che la fine delle linee 230, 240, in cui compare GO TO D il computer vada alla linea 260. Lo Spectrum é molto flessibile con i GO TO prodotti da calcoli; così potreste (anche se non servirebbe a niente) terminare le linee 230-240 con GO TO 2*130 (2 volte 130).

230 Se B e C sono uguali il computer si accorge di aver fatto la vostra stessa scelta, quindi stampa: "È UN PAREGGIO!" e poi va alla linea 260 (GO TO D, con D = 260).

240 Questa linea apparentemente complicata determina se le combinazioni di B e C assegnano la vittoria al calcolatore. Se è così il calcolatore stampa "HO VINTO!" e aggiunge 1 al suo punteggio: (LET COMP=COMP+1).). Sebbene questa frase risulti un po' strana se confrontata con l'aritmetica normale, essa significa in realtà: rendi la variabile alla sinistra del segno di =, uguale al vecchio valore della variabile, aumentato di 1.
Non preoccupatevi se non lo capite subito, perchè ciò si renderà chiaro gradualmente non appena userete un po' più spesso questo tipo di frasi.

250 Questa linea controlla le condizioni di una vostra vittoria.

260 Questa linea stampa il risultato. Notate che c'é un apostrofo prima delle virgolette e due apostrofi dopo le virgolette che seguono PUNTEGGIO e prima delle virgolette che precedono 10. Come avrete senza dubbio visto facendo partire il programma ciò mette due linee vuote prima che la linea con 10 venga

stampata. La freccetta che vedete è il simbolo di "maggiore di" e si ottiene premendo il tasto T con SYMBOL SHIFT tenuto premuto.

270 Questa PAUSE é un ritardo di 2 secondi e vi dà il tempo di leggere il risultato parziale.

280 NEXT A è la fine del ciclo FOR/NEXT iniziato alla linea 50. Come già spiegato prima questa linea rimanda l'azione alla linea 50 dove il valore di A é incrementato di uno e quindi le linee seguenti vengono eseguite nuovamente.

290 Se il valore della variabile HUM é lo stesso della variabile COMP lo Spectrum capisce che la gara é terminata con un pareggio e stampa il messaggio "LA GARA É FINITA IN PAREGGIO!" saltando due linee.

300 Se HUM é minore di COMP il calcolatore si rende conto che ha vinto e lo stampa.

310 Se COMP é minore di HUM avete vinto voi e il calcolatore lo stampa.

Spero che questa spiegazione non sembri troppo complessa. Proverò a spiegarvi i concetti fondamentali della programmazione dello Spectrum in BASIC nel modo più semplice possibile e così si potrà fare molto lavoro in poco tempo.

AGGIUNGERE UN PO' DI COLORE

Esamineremo a fondo le capacità di colore (e suono) dello Spectrum un po' più avanti nel libro, ma adesso vorrei presentarvi alcuni dei metodi più semplici per usare il colore così che possiate rendervi conto di come possa essere usato per valorizzare i programmi.

Segue una nuova versione a colori di SASSO, FORBICI e CARTA.

Potete facilmente modificare il programma usando le possibilità di correzione dello Spectrum.

```
  10 REM SASSO, FORBICI, CARTA
  20 REM (c) HARTNELL, 1982
  30 LET COMP=0
  40 LET HUM=0
  50 FOR A=1 TO 10
  60 CLS : PRINT INK 2;"GARA NUME
RO "; INK 1;A
  70 PRINT INK 1;'"1 - SASSO"
  80 PRINT INK 2;"2 - FORBICI"
  90 PRINT INK 3;"3 - CARTA"
 100 PRINT INK 6; PAPER 2;'"INTRO
DUCI 1, 2  O 3"
 110 INPUT B
```

```
 120 PRINT INK B;'"HAI SCELTO ";
 130 IF B=1 THEN PRINT "SASSO"
 140 IF B=2 THEN PRINT "FORBICI"
 150 IF B=3 THEN PRINT "CARTA"
 160 PAUSE 50
 170 LET C=INT (RND*3)+1
 180 PRINT INK C;'"IO HO SCELTO "
;
 190 IF C=1 THEN PRINT "SASSO"
 200 IF C=2 THEN PRINT "FORBICI"
 210 IF C=3 THEN PRINT "CARTA"
 220 LET D=260
 230 IF B=C THEN PRINT FLASH 1;'
"E' UN PAREGGIO!": GO TO D.
 240 IF C=1 AND B=2 OR C=2
AND B=3 OR C=3 AND B=1 THEN
PRINT FLASH 1; BRIGHT 1;'"HO
 VINTO!": LET COMP=COMP+1: GO TO
D
 250 IF B=1 AND C=2 OR B=2 AND C=
3 OR B=3 AND C=1 THEN PRINT FLASH
 1;'"HAI VINTO!": LET HUM=HUM+1
 260 PRINT '"PUNTEGGIO"'"IO> ";C
OMP;TAB 10;"TU> ";HUM
 270 PAUSE 100
 280 NEXT A
 285 INVERSE 1
 290 IF HUM=COMP THEN PRINT ''"LA
GARA E' FINITA IN PAREGGIO!"
 300 IF HUM<COMP THEN PRINT ''"HO
VINTO QUESTA PARTITA!"
 310 IF COMP<HUM THEN PRINT ''"HA
I VINTO QUESTA PARTITA!"
```

Avrete già visto che la linea 60 comprende ora anche la parola INK 2. Per inserirla anche nella vostra versione è sufficiente che premiate LIST (tasto K) quindi 60, e poi ENTER.

A questo punto apparirà la parola "scroll" in basso sullo schermo. Premete N in modo da fermare il listato del programma, poi tenendo premuto CAPS SHIFT, premete il tasto 1 (EDIT). La linea a cui il cursore (un segno >) sta puntando (in questo caso la 60) verrà visualizzata in basso sullo schermo. Vogliamo inserire nel testo le parole INK 2 dopo la parola PRINT, per cui mantenendo premuto CAPS

SHIFT, premete il tasto 8. Vedrete muoversi il cursore nella direzione della freccia del tasto 8.

Appena il cursore ha superato PRINT, ma prima che vada oltre, rilasciate 8. A questo punto premete entrambi i tasti di SHIFT poi rilasciate il CAPS SHIFT. Sempre mantenendo premuto il SYMBOL SHIFT premete il tasto X e se tutto va bene dovrebbe apparire la parola INK.

Introducete 2 e un punto e virgola (;) (SYMBOL SHIFT e tasto 0) e poi finalmente ENTER.

Dovreste vedere la nuova linea modificata prendere il posto della precedente linea 60.

Fate girare il programma con RUN e vedrete la parola GARA NUMERO 1 in rosso. I colori sono scelti con i numeri dallo 0 al 7 (come sarà illustrato più dettagliatamente in seguito). INK 1 colorerà le lettere che seguono in blu, 2 in rosso e così via. (Non c'é bisogno che facciate girare tutto il programma prima di continuare a correggere; sarà sufficiente introdurre una lettera, ad esempio una Q, quando vi é stato chiesto un numero perchè il programma si arresti stampando un messaggio di errore; lo Spectrum non capirà infatti cosa significa Q).

Premete ancora EDIT (CAPS SHIFT e 1) e la linea 60 riapparirà in fondo allo schermo. Usando la combinazione CAPS SHIFT e 8 muovete il cursore attraverso lo schermo sino a dopo le virgolette che seguono NUMERO, aggiungete INK 1 e un altro punto e virgola (che deve sempre seguire questi comandi speciali in una frase PRINT) prima della A. Premete ENTER per reintrodurre la linea e fate girare nuovamente il programma.

Dovreste vedere le parole GARA NUMERO in rosso e il numero 1 in blu. Se i colori appaiono poco chiari regolateli attraverso i controlli della sintonia e del colore della vostra televisione sino a quando il rosso e il blu non appaiono correttamente. Ogni televisore riproduce differentemente i colori e quindi potreste trovare che un colore è più intenso degli altri.

Adesso potete passare l'intero programma e fare le necessarie correzioni per aggiungere l'effetto colore. Le linee da modificare sono le seguenti: 60, 70, 80, 90, 100, 120, 180, 230, 240, 250, e 285 (quest'ultima deve essere aggiunta... penserà il calcolatore ad inserirla nell'ordine corretto).

Fate girare il programma e osservate come il colore migliori il programma e gli dia un tono diverso.

ISTRUTTORE DI DATTILOGRAFIA

L'ultimo programma di questo capitolo introduttivo é stato pensato per aiutarvi a prendere confidenza con la tastiera. Quando farete girare il programma il calcolatore metterà una lettera dell'alfabeto oppure un numero scelti a caso sullo schermo, approssimativamente nella posizione in cui si trova sulla tastiera. Avrete un tempo limitato entro il quale premere il tasto designato. Se sbaglierete il computer magna-

nimo vi concederà un'altra prova. Vi meraviglierete di quanto bene e quanto velocemente imparerete a destreggiarvi sulla tastiera.

Se credete che il tempo di risposta al gioco sia troppo corto ambiate il 100 che compare alla linea 180 con un numero più grande. Cominciate con 300 e decrementatelo gradatamente fino a trovare il tempo giusto per la vostra attuale velocità di battitura. Fate attenzione a inserire correttamente i numeri e le lettere nelle frasi DATA (le linee da 80 a 110), dato che queste controllano i vari numeri e le lettere che verranno stampati. Dopo che avete fatto girare il programma diverse volte tornate al libro per un'analisi linea per linea.

```
  10 REM ISTRUTTORE DI DATTILOGRA
FIA
  20 DIM A$(36,5)
  30 LET SCORE=0
  40 FOR A=1 TO 36
  50 READ B$
  60 LET A$(A)=B$
  70 NEXT A
  80 DATA "10502","20605","30508
","40611","50614","60617","70620
","80628","90626","00629"
  90 DATA "A1202","B1517","C1511
","D1208","E0908","F1211","G1214
","H1217","I0923","J1220"
 100 DATA "K1223","L1226","M1523
","N1520","O0926","P0929","Q0902
","R0911","S1205","T0914"
 110 DATA "U0920","V1514","W0905
","X1508","Y0917","Z1505"
 120 REM
 130 REM   PROVA
 140 REM
 150 FOR A=1 TO 10: BEEP .02,-10
: CLS
 155 IF INKEY$<>"" THEN GO TO 15
5
 160 LET B=INT (RND*36)+1
 170 PRINT FLASH 1; PAPER 6;AT V
AL (A$(B)(2 TO 3)),VAL (A$(B)(4
TO ));A$(B,1)
 180 FOR C=1 TO 100
 190 LET C=INKEY$
```

```
 200 IF C$<>"" THEN GO TO 260
 210 NEXT C
 220 PRINT AT 0,0; INC 7; PAPER
4; FLASH 1; BRIGHT 1;"AVETE IMPI
EGATO TROPPO TEMPO"
 225 FOR D=1 TO 70: NEXT D
 230 NEXT A
 240 GO TO 300
 260 IF C$=A$(B,1) THEN LET SCOR
E=SCORE+1: BEEP .25,SCORE*4: PRI
NT AT 16,0; PAPER 6; INK 2;"BRAV
O IL TUO PUNTEGGIO E' ORA ";SCOR
E'
 265 IF C$<>A$(B,1) THEN PRINT A
T 0,0; INK 1; PAPER 6; FLASH 1;"
SPIACENTE MA HAI SBAGLIATO";TAB
10;"PROVA ANCORA": GO TO 170
 270 FOR D=1 TO 100
 280 NEXT D
 290 NEXT A
 300 CLS : PRINT AT 8,2; FLASH 1
; BRIGHT 1; PAPER 6; INK 2;"IL T
UO PUNTEGGIO E' STATO DI ";SCORE
;" SU 10"
 310 IF SCORE>7 THEN PRINT AT 10
,7; FLASH 1; INK 4; PAPER 7; BRI
GHT 1;"CONGRATULAZIONI!"
```

10 Questa è una frase REM (ricordatevi che REM sta per commento) e quindi il computer la ignora. Notate che potete mettere quanti spazi volete in una frase REM.

20 La linea 20 costruisce, o meglio dimensiona, una matrice. Una matrice è usata quando volete memorizzare una lista di numeri o parole e riferirvi all'elemento della lista indicando soltanto la posizione che esso occupa nella lista. Questa matrice viene chiamata A$. Il segno di dollaro significa che gli elementi memorizzati sono parole comprese tra le virgolette e non sono numeri. Il primo elemento della matrice viene indicato con A$(1), il secondo con A$(2) e così via.

30 Pone una variabile chiamata SCORE uguale a zero. Questa variabile terrà il punteggio dei numeri e delle lettere che troverete.

40-70 Questo è un ciclo FOR/NEXT come nel programma precedente. La linea 50 (READ B$) agisce in congiunzione con le frasi DATA (le linee da 80 a 110). Legge ciascuna di esse a turno ogni volta che il ciclo viene eseguito. La prima volta legge il primo elemento nella linea DATA ("10502") e lo assegna a B$, quindi la linea 60 pone A$(1) uguale al valore di B$. La volta successiva durante il ciclo A è uguale a 2, così la linea 60 esegue LET A$(2)=B$ ma questa volta B$ ha valore del secondo elemento nella frase DATA ("20605"). Il ciclo si ripete finchè a tutti i 36 elementi della matrice A$ è stato assegnato uno dei valori delle frasi DATA.

80-110 Le frasi DATA. Notate che gli elementi sono racchiusi dalle virgolette e separati da virgole.

120-140 Tre frasi REM.

150 Questa linea inizializza un altro ciclo FOR/NEXT. I cicli FOR/NEXT non devono per forza essere collegati alla lettera A: qualsiasi lettera dell'alfabeto è accettabile, quindi FOR J=1 To10 e FOR M=1 TO 10 fanno la stessa cosa. BBEP .02,-10 produce un breve suono (BBEP sarà analizzato dettagliatamente in seguito) e CLS pulisce lo schermo.

155 INKEY$ (che troverete sopra il tasto N) legge la tastiera, aspettando che venga premuto il tasto. Lo Spectrum aspetta che il tasto venga rilasciato prima di continuare.

160 Questa linea sceglie un numero casuale compreso tra 1 e 36 per selezionare uno dei numeri da 0 a 9 o una delle lettere dell'alfabeto inglese.

170 Questa è una linea alquanto complessa; essa stampa il numero o la lettera contenuta nella locazione A$(B,1) nella sua corretta posizione. Ciò viene fatto dalla PRINT AT relativa al secondo e terzo numero della frase DATA (se ad esempio il secondo e il terzo numero sono 1 e 2 essa stamperà alla dodicesima linea) e al quarto e quinto (così se il quarto e il quinto numero nella frase DATA sono 0 e 5 stamperà dopo 5 spazi dal margine sinistro). In altre parole ciascuna delle frasi DATA contiene come primo elemento il carattere da stampare (lettera o numero che sia) e i successivi quattro elementi sono le sue coordinate sullo schermo, il secondo e il terzo numero sono la coordinata verticale e il quarto e il quinto quella orizzontale.

180-210 Questo ciclo FOR/NEXT attende per un tempo determinato la pressione di un tasto. La lunghezza di questo ritardo dipende dal numero alla fine della linea 180. Se un tasto viene premuto la linea 200 lo rileva e rimanda l'azione alla linea 260.

220 Se non premete un tasto in tempo, il calcolatore stampa il messaggio "AVETE IMPIEGATO TROPPO TEMPO".

255 Questo è un breve ritardo che viene realizzato con un ciclo FOR/NEXT. Per ottenere un ritardo potete usare sia una PAUSE che un ciclo FOR/NEXT vuoto.

230 Questa linea rimanda l'azione alla linea 150 per il prossimo test.

240 Alla fine dei 10 tentativi questa linea manda il controllo del computer alla linea 300 per stampare il risultato finale.

260 Questa linea controlla se il tasto premuto è lo stesso di quello stampato sullo schermo, se è così incrementa il vostro punteggio e stampa "BRAVO IL TUO PUNTEGGIO È ORA 3" o qualsiasi altro punteggio voi abbiate in quel momento.

265 Se il tasto premuto non era quello mostrato sullo schermo stampa "SPIACENTE MA HAI SBAGLIATO PROVA ANCORA", quindi l'azione ritorna alla linea 170 permettendovi di provare ancora.

270 Questo è un ritardo che permette la visualizzazione del numero o della lettera successiva.

290 Rimanda l'azione alla linea 150.

300 Stampa del risultato.

310 Aggiunge un messaggio di congratulazioni se il punteggio è maggiore di 7.

Troverete che questo programma è molto utile per aiutarvi a familiarizzare con la tastiera.

Il contenuto di questo capitolo è molto condensato ma spero che una lettura attenta e l'uso dei programmi al momento giusto vi abbiano dato una buona introduzione di programmazione in BASIC sullo Spectrum. Questa infarinatura dovrebbe completare quella già data dai due manuali che vi sono stati dati con lo Spectrum per aiutarvi a diventare rapidamente un efficiente programmatore. Il resto di questo libro partirà con il concetto che abbiate assorbito le informazioni date in questo capitolo e che abbiate capito il manuale introduttivo e molte delle prime cose del secondo manuale forniti con il vostro computer.

CAPITOLO 2

ESPLORIAMO IL COLORE DELLO SPECTRUM

Lo Spectrum ha sei colori (blu scuro, rosso, magenta, verde azzurro e giallo)) oltre a bianco e nero. Ogni colore è identificato da un numero, compreso tra 0 e 7. Le relazioni tra numeri e colori sono le seguenti:

0 – Nero
1 – Blu scuro
2 – Rosso
3 – Magenta
4 – Verde
5 – Azzurro
6 – Giallo
7 – Bianco

Se avete un televisore in bianco e nero o il televisore a colori con il controllo del colore completamente abbassato vedrete questi colori come una fascia di grigi che vanno dal nero al bianco in quest'ordine.

Fate questa prova:

```
10 FOR X = 0 TO 7
20 PRINT INK X; "■■" (due quadrati pieni ottenibili premendo il tasto di
   shift bianco seguito dal tasto 9 e poi dal tasto 8).
30 NEXT X.
```

I codici dei colori sono legati a come il televisore riproduce i colori.

Un televisore a colori usa il blu, il verde e il rosso per fare anche tutti gli altri mischiando opportunamente i tre di base. Il colore magenta viene fatto mischiando blu e rosso e il suo codice (3), è la somma dei codici di blu (1) e rosso (2).

Quanto accendete il vostro Spectrum potete vedere che l'interno dello schermo è bianco e che state scrivendo in nero. È facile aggiungere una bordo di colore usando il comando BORDER che si trova sul tasto B. Provate a scrivere:

BORDER 1 (seguito da ENTER).

Questo cambia il bordo dello schermo in blu scuro. Potete mettere uno qualsiasi degli otto numeri dopo il comando BORDER, e dopo aver premuto ENTER il bordo cambierà colore, secondo il comando dato. Potete anche scegliere il colore di INK e PAPER (letteralmente inchiostro e carta). Il colore che darete a PAPER sarà quello della pagina rettangolare compresa dentro il bordo e il colore INK sarà quello del testo e dei caratteri grafici che introdurrete. Ottenere i comandi PAPER e INK è un po' più difficile di quanto avete fatto per BORDER.

Per cambiare colore a PAPER dovete prima premere il tasto shift bianco sulla sinistra della tastiera, poi quello rosso sulla destra e finalmente tenendo sempre premuto lo shift rosso premere il tasto C.

Provate qualcuna delle frasi che seguono (e notate che come precedentemente con BORDER non avete bisogno di numeri di linea per questi esempi):

```
PAPER 3: CLS (troverete questo comando sul tasto V)
PAPER 2: CLS
PAPER 6: CLS
```

Vedrete il rettangolo principale dello schermo diventare prima magenta, poi rosso e poi giallo. Come probabilmente sapete potete usare uno qualunque degli otto colori dopo un comando PAPER, ma per calcolare tutta la pagina dello schermo dovrete far seguire il comando PAPER da un comando CLS (c'è un modo per evitare questo ma ci torneremo in seguito).

Il comando INK è altrettanto facile da usare. Per avere INK premete CAPS SHIFT e continuando a premerlo premete SYMBOL SHIFT sulla destra e tenendo ancora premuto lo shift rosso premete il tasto X. Il comando INK è usato allo stesso modo del comando PAPER.

Per provare l'effetto del testo colorato scrivete:

```
10  PAPER 7
20  INK 3
30  CLS
40  PRINT "Prova"
```

Se provate questo programma vedrete il calcolatore stampare il testo in magenta su uno sfondo bianco. Potete mettere il testo di un colore su uno sfondo di qualunque altro colore ma dovete tenere conto che molti colori non stanno molto bene insieme.

Per capire ciò che intendo, cambiate il programma precedente con questo che segue e provatelo:

```
10  PAPER 5
20  INK 4
30  CLS
40  PRINT "Prova"
```

È molto difficile distinguere le parole. Azzurro e verde non si mischiano molto bene e scrivere il testo in verde su sfondo azzurro rende particolarmente difficile la lettura.

Potete usare i comandi colore per colorare solo una parte dello schermo invece di cambiare l'intera pagina compresa nel bordo. Per capire ciò guardate l'istruzione che segue:

```
PAPER 7: CLS : PRINT INK 5; PAPER 1; "Prova"
```

Come potete vedere, solo l'area dietro al testo era in azzurro e il testo in blu.

Ci sono quattro cose che potete fare con il colore e tutte le parole BASIC per esse si trovano sotto la fila dei tasti in basso sulla tastiera; sono FLASH, BRIGHT, OVER e INVERSE.

Le ultime due non sono legate all'uso del colore ma hanno molti altri usi quando lavorano col colore.

BRIGHT

Il comando BRIGHT è semplice da capire, è usato per far risaltare un po' di più il colore sia del testo (INK) che dello sfondo (PAPER). Usandolo è possibile rendere i colori più chiari. In alcuni casi il comando BRIGHT rende i colori più luminosi.

Questo programma mostra quello che fa BRIGHT:

```
10 FOR x=0 TO 7
20 PRINT PAPER x; BRIGHT 0; " "; BRIGHT 1; " ";
30 NEXT x
```

Questo programma stampa degli spazi (pensateli come piccoli blocchi di PAPER) a gruppi di due. Il primo è il colore normale e il secondo è lo stesso colore schiarito. Come potete vedere il comando BRIGHT è seguito da uno 0 o da un 1. Zero disattiva BRIGHT e Uno invece lo attiva.

Il prossimo programma porta allo stesso risultato, mettendo però la BRIGHT in un ciclo:

```
10 FOR x=0 TO 7
20 FOR y=0 TO 1
```

```
30 PRINT INK x; BRIGHT y; " ";
40 NEXT y
50 NEXT x
```

FLASH

Questo comando è usato circa come la BRIGHT. Anche questo è seguito da uno 0 o da un 1 che determina la messa in opera del comando. Potremmo far lampeggiare i quadrati dell'esempio precedente con questo programma:

```
10 FOR x=0 TO 7
20 FOR y=0 TO 1
30 PRINT INK x; BRIGHT y; FLASH y; " ";
40 NEXT y: NEXT x
```

Questa routine mostra che la FLASH fa lampeggiare il quadratino di stampa sullo sfondo. Se avete appena acceso il vostro calcolatore la INK è posta a INK 9. INK 9 è una frase speciale che dice allo Spectrum di usare come INK il colore che produce il miglior contrasto con il colore attuale della PAPER. Il programma che avete appena fatto girare mostra che per i quattro colori più scuri dello Spectrum viene scelto il bianco come colore per il testo (INK) e per i quattro più chiari il nero. Quindi i colori più scuri sotto l'effetto della BRIGHT lampeggiano dal loro colore al bianco e quelli più chiari dal loro colore al nero.

INK 8 è un'altra frase corretta. Essa significa che l'ultimo colore scelto per stampare il testo va conservato.

Potete usare questi due numeri anche con il comando PAPER, ottenendo lo stesso effetto. Così PAPER 8 e PAPER 9 sono frasi valide. Potete usare il numero 8 dopo i comandi BRIGHT e FLASH, ma non il 9. In tutti i casi il numero 8 significa che rimangono le condizioni precedenti.

INVERSE, OVER

Le due istruzioni rimanenti da usare quando trattate il colore sono la INVERSE e la OVER. La prima funziona più o meno come la FLASH, però scambia il colore dello sfondo con quello del testo e viceversa. Se state scrivendo in nero su sfondo bianco la INVERSE fa scrivere bianco su nero. Agisce su tutti i colori di INK e PAPER e ha una forma simile a quella della BRIGHT e della FLASH. Ancora una volta, la parola è seguita da uno 0 o da un 1 (non in funzione e in funzione). Comunque non potete usare il numero 8 dopo questa istruzione. Provate a vedere come funziona:

```
10 FOR x=0 TO 1
```

```
20 PRINT INVERSE x; INK 5; PAPER 1; "Questo programma mostra come
   lavora la INVERSE"
30 PAUSE 50
40 NEXT x
```

La OVER è una parola molto utile del BASIC; vi consente di sovrapporre un carattere sull'altro.

Ancora una volta si usa 0 (non in funzione) e 1 (in funzione). Come la INVERSE non potete usare il numero 8 dopo OVER. La OVER funziona eseguendo una operazione 'OR esclusivo' su due matrici di carattere. Questo significa che essa adagia un carattere su un altro, ma dove trova due punti neri corrispondenti stamperà un punto bianco. Per capire meglio il modo con cui la OVER procede è opportuno considerare che ogni carattere sullo schermo (ce ne stanno 32 orizzontalmente e 24 verticalmente) è fatto di 64 piccoli punti in una matrice di 8 per 8 (più o meno come una scacchiera). Quando viene chiesto allo Spectrum di sovrapporre qualcosa, esso guarda ogni punto di entrambi i caratteri e poi esamina questa tabella di verità, per decidere se usare un punto nero o bianco:

		NUOVO CARATTERE	
		punto NERO	*punto BIANCO*
CARATTERE VECCHIO	punto NERO	BIANCO	NERO
	punto BIANCO	NERO	BIANCO

Facciamo un esempio per rendere ciò più chiaro. Stampate due caratteri sul vostro schermo, la O maiuscola e lo 0: PRINT "O 0".

Ora immaginate quello che succederebbe se sovrapponeste la O sullo 0. Essendovi formati una immagine mentale di quello che vi aspettate (tenendo conto che due neri danno un bianco), provate l'effettiva sovrapposizione:

```
PRINT OVER 1, "O"; CHR$ 8; "0"
```

(Notate che CHR$ 8 è un carattere di controllo che riporta indietro il cursore di una posizione).

Dovreste ottenere una linea obliqua. Riuscite a capire perchè?

Provate a sovrapporre altri caratteri cercando prima di indovinare cosa dovrebbe succedere. In quale altro modo può essere usatala sovrapposizione? Una delle sue applicazioni è fare apparire degli oggetti davanti o dentro altri. Un semplice esempio di ciò è il programma SOVRAPPOSIZIONE.

In questo programma una normale p si muove lungo lo schermo: quando trova una p trattata con la INVERSE, crea momentaneamente un quadratino nero, poi procede lasciando la p inversa come era (ricordate che nero e bianco danno nero).

```
  5 PAPER 5: INK 1: CLS
 10 PRINT INVERSE 1;AT 10,10;"p
"
 15 FOR a=0 TO 20
 20 PRINT PAPER 1; INK 5; INVER
SE 1; OVER 1;AT 10,a;"p"
 30 PAUSE 10
 40 PRINT INVERSE 1; OVER 1;AT
10,a;"p"
 50 NEXT a
```

ATTR

In precedenza ho detto che potete usare solo un colore INK e un colore PAPER per ogni matrice di carattere. Questo perchè lo Spectrum memorizza il colore dello sfondo, quello del testo e le caratteristiche (FLASH, BRIGHT) di ogni carattere in un'area di memoria apposita (dalla locazione 22528 alla locazione 23296). Queste 768 locazioni di memoria rappresentano le 32 colonne di 24 righe dello schermo. Ognuna di queste locazioni contiene un numero compreso tra 0 e 255 che sintetizza le informazioni sul colore, sulla luminosità ecc. di ogni quadratino dello schermo. Questo numero è rintracciabile quando usate la frase del BASIC Sinclair ATTR (X,Y) dove X e Y sono le coordinate (come nella PRINT AT) del quadratino in questione.

Il numero non é semplice da interpretare finchè non avrete fatto un po' di pratica. Esso è formato da quattro numeri sommati insieme:

128 oppure 0 a seconda che stia lampeggiando o meno
64 oppure 0 se è schiarito oppure no
il codice colore dello sfondo moltiplicato per 8 e infine il codice colore del testo.

Usate la ATTR in questo modo:

PRINT AT 10,10; FLASH 1; BRIGHT 1; INK 2; PAPER 6; "S"; ATTR (10,10)

Il calcolatore stamperà una S alla decima riga della decima colonna in colore ros-

so su sfondo giallo con sia BRIGHT che FLASH funzionanti. Cosa pensate che il computer stamperà come ATTR della posizione 10,10? Dovrebbe essere: 128+64+6x8+2=242. È particolarmente facile lavorare una volta conosciuti gli attributi di un carattere. Se vi è appena stato dato il valore di ATTR (X,Y) e volete vedere il significato di questo numero, potete usare il programma calcolatore di attributi, dato più avanti.

Potreste porre una variabile "A" uguale al valore ATTR (X,Y) in un programma e fare uso dei valori ottenuti di INK, PAPER, BRIGHT, FLASH. Un buon uso della ATTR è ispezionare la natura di un carattere in una certa posizione. Per esempio, potreste voler far attraversare ad una nave spaziale lo schermo, dove è rappresentata una montagna. Disegnereste la montagna sullo schermo con un colore differente dal resto della figura, e controllando se gli attributi dei caratteri che formano la montagna sullo schermo vengono modificati o meno dalla presenza o dall'assenza dell'astronave, potreste stabilire se vi è stata una collisione oppure no.

```
  10 INPUT "ATTR?";A
  20 IF A-128<0 THEN GO TO 50
  30 LET a=a-128: PRINT "Flash A
cceso,";
  40 GO TO 60
  50 PRINT "Flash spento,",
  60 IF a-64<0 THEN GO TO 90
  70 LET A=A-64: PRINT "Bright A
cceso,",
  80 GO TO 100
  90 PRINT "Bright spento,",
 100 IF A/8=INT (A/8) THEN GO TO
 140
 120 PRINT "Paper:";INT (A/8),",
Ink:";A-(8*INT (A/8))
 130 STOP
 140 PRINT "Ink assente,Paper";A
/8
```

In questo caso la montagna sarà in una posizione conosciuta (sebbene potreste rappresentarla sulla superficie mobile di un pianeta) e perciò l'uso della ATTR non è di grosso aiuto. Ma in altri casi potreste anche non conoscere dove si trova esattamente un "ostacolo" oppure potreste essere in grado di individuare gli ostacoli solo con molte frasi IF....THEN. In queste circostanze ATTR potrebbe risultare molto utile.

DISEGNARE UN ARCOBALENO

Proviamo qualche stampa a colori in alta risoluzione. Il prossimo programma disegna un arcobaleno. Notate come il comando per disegnare un arco di cerchio sia usato per tracciare curve colorate. Notate anche che gli archi di differenti colori sono distanziati di uno spazio. Siete in grado di capire il perchè? Può esservi soltanto un colore di INK in un solo carattere, quindi se tracciate due linee colorate differenti troppo vicine, il colore della seconda linea "contaminerà" il primo, cambiando il valore di INK di quel quadratino nel suo colore.

```
   1 REM ARCOBALENO
   2 LET x=1
   3 DIM a(6): GO SUB 100
   4 LET y=1
   5 PAPER 7: BORDER 0: CLS
  10 FOR a=160 TO 250 STEP 15
  20 FOR u=1 TO 5
  30 INK a(y): PLOT a+u,0: DRAW
-a,a-80+u,PI/2
  40 NEXT u
  50 LET y=y+1
  60 IF y=7 THEN GO TO 200
  70 NEXT a
 100 LET a(1)=3: LET a(2)=1: LET
 a(3)=5: LET a(4)=4: LET a(5)=6:
 LET a(6)=2
 110 RETURN
 200 PRINT PAPER 7;AT 19,5; INK
3;"Ŝ"; INK 1;"P"; INK 5;"E"; INK
 4;"CT"; INK 6;"R"; INK 2;"UM"
```

128 COLORI

Essere in grado di stampare due colori nell'area di un singolo carattere può servire a creare 128 colori sul vostro Spectrum.

Prima dovete definire un carattere grafico che assomigli a una piccolissima scacchiera (come sul tasto A dello ZX81).

Colorando lo sfondo con un colore e il testo con un altro, farete effettivamente metà carattere con un colore, metà con l'altro. Siccome i colori sono stesi uniformemente, i due colori si mischieranno formandone un altro. Ad esempio i punti rossi e i

punti gialli mischiati in questo modo danno un bel colore arancio. Provate ora e guardate quanti differenti colori possono essere formati sullo Spectrum con questo programma.

```
  10 REM COLORI
  20 PAPER 7: BORDER 6: CLS
  30 REM SCACCHIERA COME CHR$
  40 FOR X=0 TO 6 STEP 2
  50 POKE USR "P"+X,85
  60 POKE USR "P"+X+1,170
  70 NEXT X
  80 FOR P=0 TO 7
  90 FOR I=0 TO 7
 100 FOR B=0 TO 1
 110 PRINT PAPER P; INK I; BRIGH
T B;"PP";
 120 NEXT B: NEXT I: NEXT P
 130 PAUSE 200
 140 CLS : PRINT "E ORA RIEMPIAM
O LO SCHERMO...": PAUSE 200
 150 CLS
 160 POKE 23692,100
 170 FOR P=0 TO 7
 180 FOR I=0 TO 7
 190 FOR B=0 TO 1
 200 FOR K=0 TO 31
 210 PRINT PAPER P; INK I; BRIGH
T B;"PPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPP"
 220 REM TUTTE LE 'P' SONO
     CARATTERI GRAFICI
 230 NEXT K
 240 POKE 23692,100
 250 NEXT B: NEXT I: NEXT P
 260 STOP
```

Potreste ottenere questi colori anche in un altro modo, disegando delle sbarrette sullo schermo con i comandi DRAW e PLOT, tracciando le linee in modo che vi sia un pixel (punto) di distanza tra una linea e l'altra. Disegnando il testo con un colore su uno sfondo colorato in modo differente, può essere ottenuto lo stesso effetto di quando erano stati usati i caratteri grafici.

Lo Spectrum può produrre disegni molto belli tracciando delle linee con il coman-

do DRAW, facendole 'interferire' una con l'altra. Il programma ARAZZO dà una valida dimostrazione di questo procedimento.

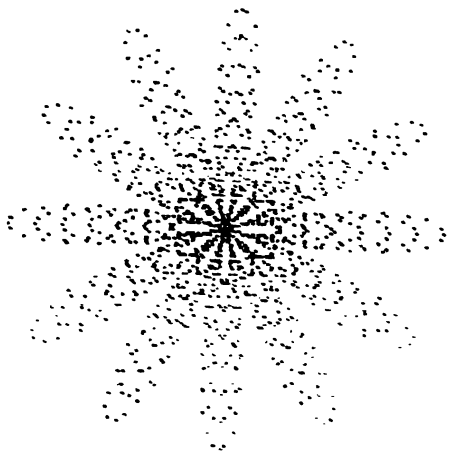
Questo programma traccia più linee dal centro dello schermo verso i bordi. Tracciate una linea da ogni punto laterale dello schermo verso il centro: otterrete una macchia uniforme di colore; ma lasciando almeno un pixel di spazio si ottiene una figura molto interessante.

Questo perchè le linee che il vostro Spectrum traccia non sono completamente diritte, ma variano a seconda del punto da cui provengono e da quello verso il quale sono dirette. Il programma disegna tutti i colori del testo in contrasto su tutti i colori possibili dello sfondo.

Qualche colore sembrerà interferire con alcuni piuttosto che con altri e altri sembreranno ruotare sullo schermo nei modi imposti dal modello.

```
  10 REM ARAZZO
  20 LET Z=1
  30 LET ink=(RND*5)+1
  40 LET paper=(RND*5)+1
  50 IF ink=paper THEN GO TO 30
  55 PAPER paper: INK ink: CLS
  60 FOR A=1 TO 2
  70 FOR X=0 TO 254 STEP 2
  80 PLOT 128,88: DRAW (-127*Z)+
(X*Z),Z*-87
  90 NEXT X
 100 FOR Y=0 TO 175 STEP 2
 110 PLOT 128,88: DRAW 127*Z,Z*-
87+(Y*Z)
 120 NEXT Y
 130 LET Z=-Z
 140 NEXT A: PAUSE 100: GO TO 20
```

Il prossimo programma è attraente almeno quanto il precedente. L'ho chiamato FIOCCO DI NEVE. Esso disegna in bianco una stella a molte punte e quindi usando una tecnica che aggiunge una nuova dimensione al vostro Spectrum, sovrappone un nuovo colore di sfondo al fiocco di neve. Naturalmente, come vi ricorderete, per cambiare il colore dello sfondo dovete prima stabilire il colore, assegnandogli il numero, e successivamente usare il comando CLS (pulitura dello schermo). Questo però distruggerebbe il modello che avete disegnato, perciò in questo programma lo Spectrum crea una stringa di spazi delle dimensioni dello schermo (704 caratteri) e la stampa sopra la figura in colori casuali ottenendo uno splendido effetto.



```
  10 REM FIOCCO DI NEVE
  20 INK 7: BORDER 3: PAPER 1: C
LS
  30 LET P$=" ": LET T$=" "
  40 FOR X=0 TO 702
  50 LET P$=P$+T$
  60 PRINT ".";: NEXT X
  70 REM CREATA STRINGA DELLE DI
MENSIONI DEL VIDEO
  80 CLS
  90 LET FLAG=0
 100 FOR K=20 TO 80 STEP 10
 110 LET Y=0: LET Z=PI
 120 LET C=100
 130 LET E=(Z-Y)/C
 140 FOR L=Y TO Z STEP E
 150 LET S=K*COS (L*6)
 160 LET H=S*SIN L
 170 LET V=S*COS L
 180 PLOT 128+H,88+V
 190 NEXT L
 200 LET FLAG=FLAG+1
 210 IF FLAG=101 THEN GO TO 230
 220 GO TO 150
 230 LET FLAG=0: NEXT K
 240 LET COL=INT (RND*6)+1
```

```
 250 PRINT AT 0,0; PAPER COL; OV
ER 1;P$
 260 PAUSE 100: GO TO 240
```

Potreste raggiungere lo stesso risultato anche in un altro modo. Riuscite a immaginare come?

La più ovvia alternativa è di inserire (col comando POKE) nell'area di memoria degli ATTRibuti i codici per cambiare il colore dello sfondo di ogni locazione dello schermo. L'area di ATTRibuti inizia alla locazione di memoria 22528 e lo schermo sul quale il colore di sfondo è stato cambiato (escludendo le due linee in basso dove vengono introdotte le nuove linee di programma) finisce alla 23231. Anche il seguente programma cambia il colore dello sfondo senza distruggere il testo o il disegno precedente:

```
10 FOR X = 22528 TO 23231
20 POKE X, 32
30 NEXT X
```

Riuscite a capire che colore sto inserendo? Ricordate che il colore dello sfondo (PAPER) è memorizzato con 8 volte il suo codice; questo programma ha stabilito come colore sfondo il 4 cioè il verde.

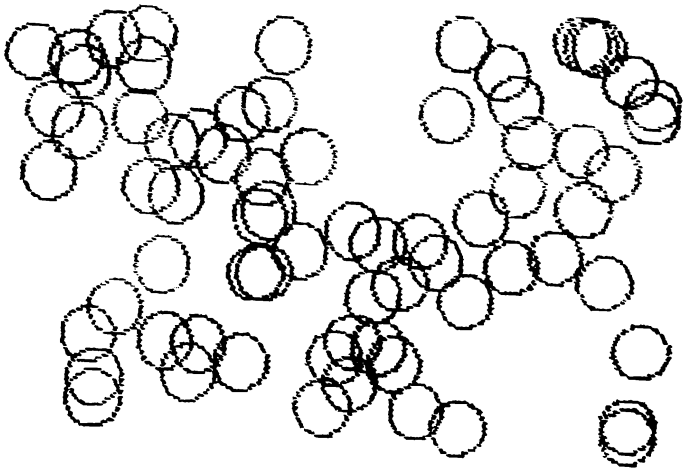
Il prossimo programma, OROLOGIO, disegna una figura molto elegante, usando molti dei comandi-colore di cui abbiamo discusso in questo capitolo.

```
  10 REM OROLOGIO
  20 PAPER 0: INK 7: BORDER 0: C
LS
  30 LET A=INT (127*RND)
  40 LET B=INT (87*RND)
  50 LET Z=INT (8*RND)
  60 LET R=INT (RND*6)
  70 IF R<2.5 THEN GO TO 140
  80 INK Z
  90 PLOT 127-A,87+B
 100 PLOT 127+A,87-B
 110 PLOT 127-A,87-B
 120 PLOT 127+A,87+B
 130 GO TO 30
 140 PLOT OVER 1;127-A,87-B
 150 PLOT OVER 1;127+A,87+B
 160 PLOT OVER 1;127-A,87+B
```

```
170 PLOT OVER 1;127+A,87-B
180 GO TO 30
```

Tracciare cerchi colorati sul vostro Spectrum può essere molto divertente.

La frase che userete è della forma: CIRCLE INK i; x, y, z dove i è il colore del testo (INK), x e y sono le coordinate del centro del cerchio, e z il suo raggio. CERCHI CASUALI è un programma che come suggerisce il nome, disegna cerchi casualmente posizionati e casualmente colorati.

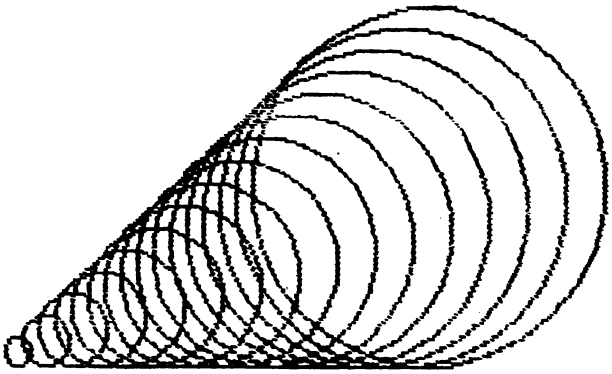


```
 1 REM CERCHI CASUALI
 5 PAPER 0: CLS
10 LET X=(234*RND)+10
20 LET Y=(154*RND)+10
30 LET Z=(5*RND)+1
40 CIRCLE INK Z;X,Y,10
50 GO TO 10
```

Il programma BOCCIA ROTOLANTE disegna cerchi concentrici di raggio crescente (usando un incremento di raggio di 0.1). Questo produce lo strano effetto di lasciare 5 aree bianche nelle BOCCE, e dentro il nome del programma.

```
 5 INK 1: PAPER 6: CLS
10 LET X=100: LET Y=X
20 FOR a=0 TO 22 STEP .1
30 CIRCLE X,Y,a
40 NEXT a
50 LET X=130: LET Y=85
60 GO TO 20
```

Disegnare i cerchi uno sull'altro, come avete avuto modo di vedere nel programma sui cerchi casuali, può produrre cerchi multicolori, perchè se due cerchi sono stampati l'uno vicino all'altro il secondo cambierà parte del colore del primo. Questo effetto é usato ancora nel programma CONO.



```
 1 REM CONO
10 BORDER 6
20 PAPER 0: CLS
30 LET Y=30
40 FOR X=30 TO 180 STEP 10
50 LET Z=(6*RND)+1
60 CIRCLE INK Z;X,Y,Y-25
70 LET Y=Y+4
80 NEXT X
90 PAUSE 0
```

Come potete vedere, i cerchi sono disegnati in dimensioni sempre maggiori e in colori casuali producendo un effetto quasi tridimensionale.

L'ultimo programma potrebbe fare da base ad un programma per disegnare e colorare figure più complesse. Esso vi permette di tracciare un triangolo di qualsiasi dimensione che stia però nei bordi dello schermo e di colorare il suo interno col colore che volete.

```
   1 REM DISEGNATORE DI TRIANGOL
I
   5 BORDER 1: PAPER 7: CLS
  10 INPUT "COORDINATA X DI PART
ENZA";X
  20 INPUT "COORDINATA Y DI PART
ENZA?";Y
  30 INPUT "ALTEZZA?";H
  35 IF Y+H>175 THEN GO TO 100
  40 INPUT "LUNGHEZZA?";L
  45 IF X+L>255 THEN GO TO 100
  50 INPUT "COLORE?";C
  55 CLS
  60 FOR P=0 TO H
  70 PLOT X,Y: DRAW INK C;L,P
  80 NEXT P
  90 STOP
 100 CLS : PRINT "ESCI DAL BORDO
"''"INTRODUCI ANCORA"
 110 GO TO 10
```

CAPITOLO 3

ESPLORIAMO IL SUONO DELLO SPECTRUM

Come probabilmente sapete é molto semplice ottenere dei suoni dal vostro Spectrum. Dovete usare il comando BEEP (lo troverete stampato in rosso sulla tastiera sotto il tasto Z).

Per ottenerlo premete il tasto CAPS SHIFT bianco e il SYMBOL SHIFT assieme e poi, sempre tenendo premuto lo shift rosso premete il tasto Z. Per produrre un suono, scrivete sulla tastiera: BEEP 1,0 (seguito da ENTER). Se provate dovreste sentire una nota lunga circa un secondo e di tonalità corrispondente a un DO medio. Il primo numero si riferisce a quanto deve durare la nota, mentre il secondo si riferisce alla tonalità. La durata, che deve essere compresa tra 0.00125 e 10, è espressa in secondi. Qualsiasi tempo più corto di 0.00125 non produrrà alcun suono; con un tempo più lungo di 10 secondi verrà stampato un messaggio di errore: "Integer out of range" (fuori dalla fascia consentita).

Il secondo numero deve essere compreso tra -60 e 69, ogni gradino indica un semi tono sopra o sotto il DO medio (che è rappresentato dal numero 0). Quindi BEEP 1,1 produrrà una nota di un semitono più alta del DO medio (un DO diesis) per un secondo, e BEEP 1,-10 produrrà un suono 10 semitoni sotto il DO medio (un RE).

Ricordatevi di mettere la virgola tra i due numeri.

Potrete farvi un'idea della fascia di tonalità prodotta dallo Spectrum facendo girare il seguente programma:

```
10 FOR n=-60 TO 69
20 BEEP .2,n
30 NEXT n
```

Come sentite le note variano da veloci colpi secchi a trilli molto alti. Come potete notare, la fascia più utile per un uso musicale è tra -20 e 20, e questo spiega la scelta del DO come posizione 0.

Comunque il tono più alto e quello più basso possono risultare utili per altri scopi, come vedremo in seguito.

Vi sarete probabilmente accorti di quanto sia basso il suono. Per ottenere un suono più forte, dovete collegare lo Spectrum ad un amplificatore esterno. La maggior parte degli amplificatori con una presa "mic" é utilizzabile. Potete prelevare il se-

gnale sia dalla presa "mic" che dalla "ear" del vostro Spectrum (sono sul retro). Il segnale proveniente dalla presa "ear" è leggermente più forte. Dovreste sperimentare entrambi i casi per decidere qual'é il migliore da usare con il vostro amplificatore.

Una volta attaccato l'amplificatore esterno dovreste avvertire un "click" ogni volta che premete un tasto. Questa retroazione quando viene premuto un tasto potrebbe essere utile, ma il rumore prodotto senza amplificatore esterno sarebbe troppo basso per essere di qualche utilità. A ciò potrete rimediare introducendo il seguente comando diretto: POKE 23609,100 (poi premete ENTER).

Ora ogni volta che premete un tasto vi sarà un suono distinto. Questo potrebbe essere molto utile quando scrivete velocemente, osservando lo schermo solo a tratti. Comunque questo suono verrà amplificato ulteriormente anche dal vostro amplificatore esterno. Potete aumentare ancora il volume con l'istruzione precedente o lasciare il vostro Spectrum nel suo stato originale. Naturalmente dovrete anche staccare e riattaccare la spina ogni volta che avete bisogno di una delle prese e questo sarebbe una grossa perdita di tempo. Se ad esempio usate un amplificatore esterno, dovrete staccare l'amplificatore dallo Spectrum ogni volta che salvate o caricate un programma (a seconda di quale uscita avete sfruttato).

Ma torniamo alla musica. Dato che ogni numero di nota corrisponde alla salita di un semitono (o alla discesa se il numero è negativo) e dato che 12 semitoni fanno un'ottava, il DO medio è il numero 0, il DO successivo è il numero 12 e così via.

Potreste scrivere musica sullo Spectrum mettendo le vostre note sul pentagramma, e quindi convertire ogni nota in una frase BEEP con i relativi numeri di nota e durata (potreste fissare la nota intera alla durata di un quarto di secondo). Ma ciò è uno spreco di tempo. Sarebbe molto più comodo essere in grado di scrivere una melodia come una serie di lettere e di numeri (per la durata di ogni singola nota).

Il nostro programma MUSIC PLAYER vi permette di fare questo. Esso vi dirà di trattare le lettere maiuscole dalla A alla G come l'ottava più bassa (si ricorda che la nomenclatura delle note nei paesi anglosassoni é diversa da quella italiana in quanto C é il nostro DO, D il RE e così via) e le lettere dalla a alla c (minuscole) come le note dell'ottava successiva. Introducete la vostra melodia come una singola linea dove ogni nota é seguita da un numero che indica la sua durata. Ho scelto 1 per rappresentare una nota di una battuta, 2 per due battute e così via. Per avere una possibilità di scelta sulla velocità di queste battute aggiungete al programma questa linea:

```
5 INPUT "Che velocità scegli? 1 é lento 5 è veloce"; X
```

Se avete aggiunto la linea precedente dovrete cambiare la 370 dove è eseguito il comando BEEP:

```
370 BEEP X(Z) /X, Y(Z)
```

Come vedrete dopo aver provato questo programma è molto più facile scrivere la musica così, piuttosto che scriverla precedentemente sul pentagramma.

```
  10 REM MUSIC PLAYER
  20 DIM X(50): DIM Y(50)
  30 LET K=0: LET L=1
  40 BORDER 2: PAPER 4: INK 9: C
LS
  50 PRINT AT 0,10; INVERSE 1;"M
USIC PLAYER"''
  60 PRINT "INTRODUCETE LA VOSTR
A MUSICA    COME UNA LINEA DI NU
MERI E DI LETTERE."
  70 PRINT ''"METTETE LE NOTE CO
ME LETTERE    SEGUITE DALLA LORO
 LUNGHEZZA    IN BATTUTE"
  80 PRINT ''"AVETE DUE OTTAVE,L
A PIU' BASSA VA DA A A G E LA PI
U' ALTA    DA a A c."
  90 INPUT N$
 100 FOR A=1 TO LEN N$ STEP 2
 110 IF CODE N$(A)<97 THEN GO TO
 280
 120 IF N$(A)="a" THEN LET K=-0.
5
 130 IF N$(A)="d" THEN LET K=0.5
 140 IF N$(A)="e" THEN LET K=1
 150 IF N$(A)="f" THEN LET K=1
 160 IF N$(A)="g" THEN LET K=1.5
 170 LET Y(L)=(CODE N$(A)-87)+(2
*K)
 180 LET L=L+1
 190 LET K=0
 200 NEXT A
 210 LET L=1
 220 FOR T=2 TO LEN N$ STEP 2
 230 LET X(L)=VAL N$(T)/2
 240 LET L=L+1
 250 NEXT T
 260 GO TO 360
 270 STOP
 280 IF N$(A)="A" THEN LET K=-0.
5
 290 IF N$(A)="B" THEN LET K=-0.
5
 300 IF N$(A)="F" THEN LET K=0.5
```

```
 310 IF N$(A)="G" THEN LET K=0.5
 320 LET Y(L)=(CODE N$(A)-67-K)*
2
 330 LET K=0
 340 LET L=L+1
 350 GO TO 200
 360 FOR Z=1 TO LEN N$/2
 370 BEEP X(Z)/2,Y(Z)
 380 NEXT Z
```

PIANOFORTE

Sarebbe bello poter suonare lo Spectrum come un pianoforte. Il prossimo programma vi permette di fare proprio questo. Esso usa il comando INKEY$ per scandire la tastiera e stabilire quale tasto avete premuto. Ho scritto il programma in modo da lasciarvi una scelta di tre ottave, ma con il DO sempre sul tasto T. Sullo schermo sarà sempre presente la rappresentazione di una tastiera per aiutarvi a ricordare l'esatta corrispondenza fra tasti e note. Potete pensare i tasti da Q a P come i tasti bianchi del pianoforte e i numeri sopra di essi come i tasti neri. Lo Spectrum é stato realizzato per suonare con una scala musicale molto simile a quella di un pianoforte; potrete cosi ottenere, a volume ragionevole, una buona musica dal vostro "piano".

```
  10 REM PIANOFORTE
  20 INK 7: BORDER 0: PAPER 2: C
LS
  30 PRINT AT 5,12; INVERSE 1;"P
IANFORTE"
  40 PRINT AT 8,0;"I TASTI BIANC
HI DEL VOSTRO PIANO SONO QUELLI
CHE VANNO DA 'Q' A 'P'"
  50 PRINT AT 12,0;"PREMENDO 'Z'
 SI OTTIENE L'OTTAVA ALTA,'X' E'
' QUELLA PIU' BASSA"
  60 PRINT AT 16,0; INVERSE 1;"P
REMERE 'V' PER IL VIBRATO E 'M'
PER DISINSERIRLO"
  70 PRINT AT 19,0; INVERSE 1;"P
REMENDO 'C' SI PREDISPONE IL DO
MEDIO ANCORA SUL TASTO 'T'"
  80 LET K=0: LET X=0.3
```

```
  90 PAUSE 500
 100 REM ESPOSIZIONE VISIVA
 110 CLS
 120 PRINT ; INVERSE 1;AT 10,4;"
Q";AT 10,6;"W";AT 10,8;"E";AT 10
,10;"R";AT 10,12;"T";AT 10,14;"Y
";AT 10,16;"U";AT 10,18;"I";AT 1
0,20;"O";AT 10,22;"P"
 130 PRINT PAPER 0;AT 8,5;"2";AT
 8,7;"3";AT 8,9;"4";AT 8,13;"6";
AT 8,15;"7";AT 8,19;"9";AT 8,21;
"0"
 140 IF INKEY$="Z" THEN LET K=12
 150 IF INKEY$="X" THEN LET K=-1
2
 160 IF INKEY$="C" THEN LET K=0
 170 IF INKEY$="V" THEN LET X=0.
03
 180 IF INKEY$="M" THEN LET X=0.
3
 190 IF INKEY$="2" THEN BEEP X,-
6+K
 200 IF INKEY$="3" THEN BEEP X,-
4+K
 210 IF INKEY$="4" THEN BEEP X,-
2+K
 220 IF INKEY$="6" THEN BEEP X,1
+K
 230 IF INKEY$="7" THEN BEEP X,3
+K
 240 IF INKEY$="9" THEN BEEP X,6
+K
 250 IF INKEY$="0" THEN BEEP X,8
+K
 260 IF INKEY$="Q" THEN BEEP X,-
7+K
 270 IF INKEY$="W" THEN BEEP X,-
5+K
 280 IF INKEY$="E" THEN BEEP X,-
3+K
 290 IF INKEY$="R" THEN BEEP X,-
1+K
 300 IF INKEY$="T" THEN BEEP X,0
```

```
+K
 310 IF INKEY$="Y" THEN BEEP X,2
+K
 320 IF INKEY$="U" THEN BEEP X,4
+K
 330 IF INKEY$="I" THEN BEEP X,5
+K
 340 IF INKEY$="O" THEN BEEP X,7
+K
 350 IF INKEY$="P" THEN BEEP X,9
+K
 360 GO TO 130
 370 STOP
```

Il programma vi chiede l'ottava desiderata; potete scegliere il "DO" medio sul tasto "T" oppure l'ottava inferiore o quella superiore. Naturalmente nulla vi impedisce di riscrivere le linee dalla 140 alla 350, per poter produrre altre note premendo tasti differenti.

Infine il programma vi chiede se desiderate l'effetto vibrato. Se é così la durata della nota é posta a 0.03 secondi invece di 0.3.

Provate altri tempi e osservate l'effetto. Io ho trovato che tempi superiori a mezzo secondo rendono l'esecuzione troppo lenta, e tempi inferiori a tre centesimi di secondo producono un suono troppo secco. Vi potrebbe interessare la possibilità di variare il tempo. Avrete bisogno di conoscere la durata di ogni nota, che cambia per produrre varianti del tipo lento, veloce, veloce. Provate una routine con un comando GO SUB. Questa dovrà suonare la nota richiesta con la durata base e poi farla seguire con due o più mezze note (o magari quarti) lunghe come la prima. Non dimenticate che la routine richiede un certo tempo per essere eseguita così l'intero programma potrebbe avere dei tempi di risposta piuttosto lunghi se aggiungete troppi particolari.

Lo Spectrum è in grado di accettare anche valori frazionati della nota, non solo del tempo; questo potrebbe tornare comodo se volete modulare il vostro "organo-Spectrum" come un altro strumento musicale; é facile da ottenere se aggiungete le linee:

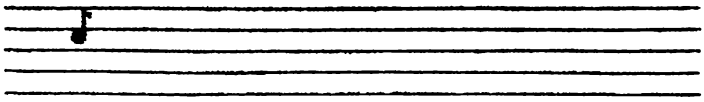
```
355 IF INKEY$="K" THEN LET K=K+0.05
356 IF INKEY$="L" THEN LET K=K-0.05.
```

Così facendo le note che suonerete con premuto uno dei tasti K o L, saranno una frazione di diesis o di bemolle (K=diesis, L=bemolle). Questo vi potrebbe servire per musiche orientaleggianti che hanno 16 o più note al posto delle consuete 12.

ALTRI EFFETTI

Potete fare di più col vostro Spectrum oltre che trasformarlo in un piano. Si può ottenere una rappresentazione visiva delle note eseguite. Per far questo adopereremo la capacità dello Spectrum di definire nuovi caratteri; questa capacità verrà spiegata più dettagliatamente in seguito nel capitolo dedicato ai giochi. Il programma MUSIC WRITER fa da base ad un programma più lungo che vi permette di scrivere la vostra musica meglio di quanto avete fatto finora, usando lettere, numeri e segni #: questa volta infatti il calcolatore mette ogni nota in una specie di "lavoro all'uncinetto" e poi stampa la nota sulla riga corretta del pentagramma.

Potete modificare questo programma in modo tale da far stampare le note sullo schermo mentre suonate lo Spectrum come un organo o un pianoforte.



```
  10 REM MUSIC WRITER
  20 PAPER 6: INK 0: BORDER 7: C
LS
  30 LET X=151
  40 REM DISEGNO PENTAGRAMMA
  50 FOR Y=1 TO 5
  60 PLOT 0,X: DRAW 255,0
  70 LET X=X-8
  80 NEXT Y
  90 FOR A=0 TO 7
 100 READ T
 110 POKE USR "L"+A,T
 120 NEXT A
 130 FOR B=0 TO 7
 140 READ T
 150 POKE USR "K"+B,T
 160 NEXT B
 170 PRINT OVER 1;AT 3,3;"L";AT
4,3;"K"
 180 BEEP 0.25,0
 190 REM PARTE SUPERIORE DELLE N
OTE
 200 DATA 0,4,7,5,4,4,4,4
```

```
 210 REM PARTE INFERIORE DELLE N
OTE
 220 DATA 0,60,124,124,124,56,0,
0
 230 STOP
```

AGGIUNGERE EFFETTI SONORI

Vi è una buona probabilità che siate interessati a realizzare dei giochi sul vostro Spectrum. Il computer é in grado di aggiungere la dimensione sonora ai giochi? Può simulare colpi laser, passi, rumori del treno in corsa?

La risposta è SI e NO. Lo Spectrum può produrre svariati suoni e rumori, utilissimi per migliorare i vostri giochi.

Tuttavia se vi aspettavate suoni o rumori buoni come quelli prodotti dai video-games, resterete alquanto delusi.

I programmi LASER, LASER 2 e LASER 3 danno una traccia per ottenere un suono apprezzabile, utilizzabile per un gioco tipo Bataglia Spaziale; CAMMINATA vi mostra come abbinare suono e movimento. Notate che alla linea 140 dovete usare i caratteri grafici L e K.

```
10 REM LASER
20 LET D=0.0125
30 FOR X=1 TO 2
40 FOR Y=4 TO 16 STEP 2
50 BEEP D,Y
60 NEXT Y
70 NEXT X
```

```
10 REM LASER 2
20 FOR X=-10 TO 0
30 BEEP 0.0125,X
40 NEXT X
50 FOR Y=0 TO -5 STEP -1
60 BEEP 0.0125,Y
70 NEXT Y
```

```
10 REM LASER 3
20 FOR X=5 TO 20 STEP 1.5
```

```
 30 BEEP .008,X
 40 NEXT X
 50 FOR Y=20 TO 5 STEP -1.5
 60 BEEP .008,Y
 70 NEXT Y
 80 PAUSE 30
 90 RUN


 10 REM CAMMINATA
 20 FOR A=0 TO 7
 30 READ X
 40 POKE USR "L"+A,X
 50 NEXT A
 60 FOR A=0 TO 7
 70 READ X
 80 POKE USR "K"+A,X
 90 NEXT A
100 FOR T=0 TO 31
110 PRINT AT 21,T;"~";
120 NEXT I
130 FOR C=0 TO 31
140 PRINT AT 20,C;"L": PAUSE 3:
 PRINT AT 20,C;"K": PAUSE 3: PRI
NT AT 20,C;" "
150 BEEP 0.02,30: BEEP 0.02,40
160 NEXT C
170 GO TO 100
180 REM DATI
190 DATA 24,36,153,126,24,100,1
32,4,24,36,153,126,24,38,33,32
200 STOP
```

Un'altra utile routine è quella che simula il suono della caduta di una bomba. Questo programma fa cadere la frequenza della nota prodotta, dal più alto valore ottenibile ad un valore di circa 20 semitoni più basso. La velocità di caduta é già elevata in partenza, ma aumenta sempre di più, per dare l'impressione della caduta della bomba.

```
10 REM BOMBA
20 FOR X=69 TO 55 STEP -0.3
```

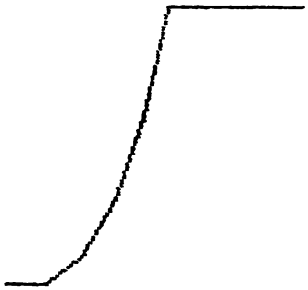
```
  30 BEEP 0.05,X
  40 NEXT X
  50 FOR Y=0 TO 20
  60 BEEP 0.01,-10: BEEP 0.01,-5
0: BEEP 0.01,-60
  70 NEXT Y
```

I micro computer più sofisticati offrono, oltre alla possibilità di suonare singole note, anche quella di produrne due o tre assieme, con variazioni di frequenza (detti sviluppi di frequenza) e talvolta possiedono un generatore di rumore bianco per creare suoni simili alla risacca del mare. Il vostro Spectrum non può certo affrontare questo confronto, ma potete usare qualche trucchetto per passare oltre i semplici suoni dati da BEEP.

FORME DI FREQUENZA E TEMPO e FORME DI FREQUENZA vi mostrano come produrre sviluppi di frequenza di vario genere.

Nel primo di questi programmi siete in grado di variare il passo fra due note e potete anche variare la durata della nota.

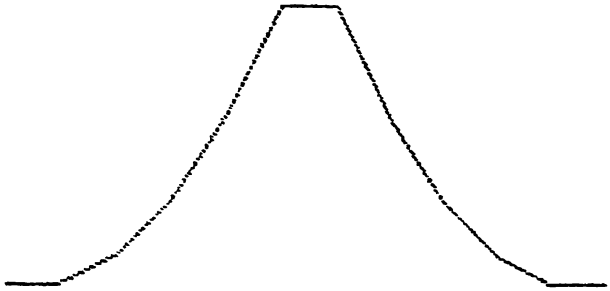
In questo esempio si va dal DO medio fino al tono più alto, salendo con un tempo di battuta piuttosto corto; il tempo diventa sempre più corto man mano che ci si avvicina alla frequenza finale.



```
  10 REM FORME DI
  15 REM FREQUENZA E TEMPO
  20 INK 7: BORDER 6: PAPER 1: C
LS
  30 LET DUR=0.05
  40 FOR F=0 TO 2 STEP 0.5
```

```
 50 BEEP DUR,F
 60 DRAW 300*DUR,20*F
 70 LET DUR=DUR-0.005
 80 NEXT F
 90 DRAW 50,0: BEEP 0.5,2
100 GO TO 100
```

Il secondo programma sale (o scende) da una frequenza data fino alla successiva, con un passo che potete definire alla linea 40. Invece di far "saltare" il suono da un valore all'altro, il programma lo fa salire verso il successivo gradatamente, incrementando il valore con salti frazionari e non interi come prima.



```
 10 REM FORME DI FREQUENZA
 20 BORDER 4: PAPER 4: CLS
 30 REM POTETE CAMBIARE
       I VALORI DI F,FE,S
 40 LET F=0: LET FE=3: LET S=1
 50 FOR X=F TO FE STEP S
 60 DRAW 25*S,20*X
 70 BEEP 0.03,X
 80 NEXT X
 90 DRAW 20,0: BEEP 0.2,FE
100 FOR Y=FE TO F STEP -S
110 DRAW 25*S,-20*Y
120 BEEP 0.03,Y
130 NEXT Y
140 GO TO 140
```

Potreste desiderare di sperimentare diversi di questi sviluppi contemporaneamente, per ottenere suoni più complessi. Facendo questo tipo di cose, però, si rende il suono ancora più basso; quindi vi raccomando caldamente l'uso di un amplificatore esterno.

L'ultimo programma di questo capitolo, maiuscolo, collega le capacità cromatiche e sonore dello Spectrum. Vengono suonate note di lunghezza e frequenza casuale (con limiti prefissati). Più è alta la nota e più in alto sullo schermo viene stampata una macchina di colore; più é lunga la nota e più verso destra viene stampata la macchia. Il risultato é un collage di colori dove quello che vedete corrisponde a ciò che sentite.

```
 10 REM MUSICGRAPH
 20 BORDER 0: PAPER 0: CLS
 30 FOR X=0 TO 31
 40 LET NOTE=RND*20
 50 LET DUR=RND*0.4
 60 LET INK=(RND*6)+1
 70 PRINT INK INK;AT 20-NOTE,X;
"■"
 80 BEEP DUR,NOTE
 90 NEXT X
100 CLS : GO TO 30
```

CAPITOLO 4

USARE LO SPECTRUM IN AFFARI

Come tutti i computer, lo Spectrum da solo non può fare niente. Con del software appropriato può essere affrontata una vasta gamma di problemi. Gli utenti comprano spesso del oftware che contiene anche caratteristiche non richieste, ma che comunque svolge il lavoro base per cui è stato acquistato.

Man mano che si va avanti (se il sistema é buono) gli utenti cominciano gradualmente a usare le caratteristiche precedentemente inutilizzate e dopo poco tempo non capiscono come abbiano fatto ad andare avanti senza di esse. Più avanti sono brevemente accennati alcuni esempi di ciò che può fare un sistema automatizzato. Questi esempi mostrano che i sistemi informativi possono fornire un aiuto enorme se sono organizzati adeguatamente.

Come ho già puntualizzato nel primo capitolo, i calcolatori possono fare qualsiasi cosa che sia riducibile ad una serie di comandi elementari eseguibili dal computer. Un computer agirà sempre in accordo con le informazioni che gli sono state fornite. Per esempio se il vostro programma di fatture contiene una sezione del tipo: "Spedire un avviso a tutti coloro che non hanno versato la rata del mese" qualche persona si vedrà arrivare a casa avvisi di pagamento per 0 Lire.

Il grado di accuratezza richiesto nel produrre e nell'usare programmi dipende dal tipo di applicazione. Lo sforzo richiesto per assicurarsi che lo Space Shuttle vada in su piuttosto che in giù sarebbe stato sprecato per il miglioramento di un programma gioco sugli anagrammi.

Le applicazioni dei calcolatori possono essere raggruppate in diverse classi (con limiti non molto definiti).

APPLICAZIONI GESTIONALI

Normalmente, operazioni molto semplici sono portate a termine su quantità relativamente grandi di dati. La gestione di magazzino, ad esempio, non richiede niente di più difficile di sottrazioni e addizioni; ma ciò che é importante é trovare il dato richiesto il più velocemente possibile. La maggior parte del tempo che occorre a far girare un programma é impiegato per gestire le informazioni contenute in memoria. Un sistema gestionale ben disegnato comprenderà unità che possono interagire l'una con

l'altra. Una vendita dovrebbe produrre dei cambiamenti nei vostri libri contabili e nelle vostre scorte; il computer dovrebbe preparare la relativa fattura e le note di avviso. Ci sono molti vantaggi ad avere un programma di gestione testi interamentee con i programmi gestionali. Tutto dovrebbe essere il più possibile a prova di errore. In particolare gli errori umani dovrebbero essere previsti e controllati, dove é possibile. Per esempio il programma dovrebbe controllare che le entrate siano in qualche modo sensate.

Questi sistimi gestionali sono piuttosto rari ma senza dubbio sono anche molto validi.

Un buon software é costoso sia da produrre che da mantenere.

Uno dei package disponibili per lo Spectrum é un programma di progettazione molto giustificato ed è considerato alla pari con altri simili programmi reperibili sul mercato. Costa molto meno di tutti gli altri programmi in grado di progettare, ma costa più di uno Spectrum da 16 K. In realtà l'utente accorto spenderà di più nel valorizzare il package che nell'acquistarlo (uno sviluppo del software potrebbe richiedere diversi giorni di lavoro di una persona altamente qualificata). Ci vorrà del tempo per dire se gli utenti saranno preparati a pagare il prezzo necessario per stimolare la produzione di software di prima qualità.

WORD PROCESSING

Con una tastiera aggiuntiva lo Spectrum è in grado di diventare un più che rispettabile Word Processor. Un Word Processor vi permette di inserire un testo e di vederlo sullo schermo. Potete inserire, cancellare o spostare il testo a vostro piacimento: alla fine potete farlo stampare nel formato voluto, poi potete anche tornare indietro e fare ulteriori modifiche.

APPLICAZIONI SCIENTIFICHE

Quando i calcoli sono molto complessi, la memoria utilizzata, gli input e gli output richiesti sono relativamente piccoli, ma è necessario molto tempo della CPU per svolgere effettivamente i calcoli.

APPLICAZIONI TECNICHE

Lo Spectrum può essere usato per semplificare la vita a persone come architetti, progettisti e ingegneri. Anche una capacità di programmazione a livello elementare può servire moltissimo in queste applicazioni, quando sono richiesti molti calcoli insoliti.

Esiste anche software tecnico standard ed è spesso di notevole valore.

APPLICAZIONI DI CONTROLLO ED ELABORAZIONE IN TEMPO REALE

In questo gruppo di applicazioni il calcolatore è collegato col mondo reale. Il risultato di un processo non sempre è una semplice stampa, ma può anche essere un'azione come spegnere o accendere un motore. In molti casi i programmi sono temporizzati: i risultati devono essere esatti e prodotti al momento giusto, nè troppo presto nè troppo tardi.

Come noterete c'è una grande differenza con la maggior parte delle altre applicazioni che possono procedere in modo indipendente.

COMUNICAZIONI

I calcolatori possono essere usati per comunicare con un gran numero di dispositivi, inclusi in particolare altri calcolatori. Ciò può essere fatto usando le normali linee telefoniche: si aprono così molte possibilità interessanti.

Lo Spectrum è più che adatto alle applicazioni gestionali perché è una macchina potente ad un prezzo molto ragionevole. Se è conveniente usare un calcolatore,esistono reali ed ovvi vantaggi nel possederne uno. Più o meno come tutti quelli che hanno usato una macchina da scrivere trovano utile possedere un Word Processor.

Lo Spectrum può risaltare come controllore di processi in tempo reale e mostrerà le sue capacità svolgendo un utile lavoro scientifico.

Di solito si raccomanda, piuttosto giustamente, di guardare al software di cui avete bisogno, prima di decidere quale calcolatore comprare. Molta gente è stata abbagliata nel comprare il calcolare da impressionanti specifiche, per scoprire poi che il software desiderato non era disponibile, o che per ottenere i risultati voluti era necessario aggiungere qualcosa di molto costoso. Questa tendenza potrebbe forse essere modificata in vista dell'enorme mercato che ha lo Spectrum. Qualsiasi cosa di cui avete bisogno è disponibile ora o lo sarà molto probabilmente in futuro. Il software gestionale dello Spectrum tende ad essere molto economico rispetto al software analogo per altre macchine. Comunque, ampie dimostrazioni, assistenza dopo la vendita e modifiche per i clienti sono molto rare. Quando selezionate il software per lo Spectrum tenete presente che non sempre la qualità del personale che produce il software e la pubblicità sono in relazione con la qualità del software stesso. Suggerimenti di altri utenti e in una certa misura le riviste specializzare sono le migliori fonti di informazione.

Anche se decidete di non comprare al momento software commerciale, ci sono molti modi di usare il calcolatore per aiutarvi a far girare programmi che avete scritto o adattato per vostro conto.

Questo capitolo è basato su cinque programmi che ho scritto per darvi un'indicazione su come lo Spectrum può aiutarvi in piccoli affari o in casa. I programmi potrebbero anche non soddisfare i vostri attuali desideri, ma dovrebbe essere piuttosto semplice modificarli per renderli adatti a voi.

I programmi sono:

Contabilità personale

Vi permette di precisare le vostre spese ricorrenti per poterne tenere conto; come gli altri programmi contiene un menù e vi permette di salvare il vostro bilancio su nastro.

Agenda

Ordina i dati cronologicamente e li stampa su richiesta.

Elenco telefonico

Accetta nomi e numeri di telefono (come avrete già capito) e ricerca rapidamente nell'elenco il nome richiesto; l'elenco può essere modificato ed espanso ogni volta.

Base di dati

Memorizza fino a 130 elementi doppi di 24 caratteri (su una macchina da 16 K; un file più grande può essere creato su una da 48 K); vi permette l'accesso ai dati secondo due criteri differenti. Il programma è stato realizzato per memorizzare una sequenza i 130 record; può stampare una lista di record in ordine alfabetico secondo l'autore o il titolo e vi permette ricerche per mezzo di ciascuna lista. Può essere adattato per trattare altri tipi di dati, e se è richiesto solo un file o se sono sufficienti meno caratteri per file, può trattare molti più dati, sempre mantenendo le capacità di ordinamento e ricerca.

Modelli finanziari/proiezioni di vendita

Uno dei più famosi package di software gestionale al mondo è il Visicalc, un grande foglio elettronico che consente all'utente di inserire dati o simboli e poi ottenere proiezioni basate su quei dati. Questo programma svolge appunto due funzioni del Visicalc: analizza l'andamento delle vendite mese per mese (o qualsiasi altro periodo di tempo desiderato), e vi consente di fare delle previsioni basate sulla media dei mesi precedenti, oppure sui risultati delle vendite dell'ultimo mese.

Questi programmi potrebbero avere bisogno di qualche modifica per esservi d'aiuto in piccoli affari o in casa. La ragione principale per averli inclusi in questo capitolo, a parte l'uso immediato, è mostrare i metodi con cui il software gestionale può essere scritto. Il software gestionale contiene raramente una codifica "intelligente" come invece avviene per moltissimi videogames delle sale giochi. L'approccio al programma è generalmente più metodico e meno eccitante. Di solito, i compi-

ti che un programma gestionale deve adempiere possono essere specificati prima che voi cominciate, mentre un gioco si evolve man mano che viene scritto.

Una volta che sapete cosa state tentando di ottenere da un programma gestionale troverete relativamente facile scrivere il programma o trovare qualcuno che lo scriva per voi.

Daremo adesso un'occhiata ai programmi, uno per uno, e vedremo ciò che possiamo imparare sullo scrivere software gestionale dai vari listati.

CONTABILITÀ

Quando per la prima volta lo farete girare vedrete apparire il seguente menù:

Il bilancio attuale è di Lit 0
Introdurre
1 — Per inizializzare i dati
2 — Per modificare la tabella di pagamento
3 — Per depositare denaro
4 — Per salvare su nastro
5 — Per finire

È una buona idea fornire un menù come questo, per assicurarsi che il programma possa essere usato senza elaborate istruzioni o con il programmatore sempre appresso all'utente per spiegargli passo passo cosa deve fare. Assumiamo ragionevolmente che quando fate girare il programma per la prima volta introduciate 1 "Per inizializzare i dati". Appena introdotto 1 vedrete "Introdurre il numero di spese che devono essere pagate ogni mese".

Allora introdurrete il numero delle vostre spese ricorrenti, come i mutui, la scadenza delle rate dell'auto, l'affitto, la carta di credito e così via.

Sarebbe meglio essere larghi con il numero di categorie, perché è impossibile aggiungerne altre quando avete inizializzato il package. Se introducete 4 come numero di spese ricorrenti, vi verrà chiesto di introdurre uno per uno i 4 "nomi" degli elementi e quindi la consistenza della spesa mensile. I nomi degli elementi e i loro costi possono essere facilmente cambiati in futuro. Una volta che avete completato la lista (inserendo spazi vuoti nelle voci che per il momento non usate), vi verrà data la possibilità di modificare qualsiasi voce volete: "Se è tutto OK premi ENTER. Se c'è un errore introduci il numeo della spesa da cambiare".

Se il programma ancora non conosce l'entità del vostro conto in banca o meglio del vostro bilancio vi chiederà: "Introduci l'ultimo bilancio noto" e quindi "Introduci le entrate una per una includendo il salario; termina le entrate introducendo ~E~".

Mano a mano che introducete ogni entrata, il bilancio in alto a sinistra sullo scherzo viene aggiornato. Una volta che avete terminato l'introduzione delle entrate inserendo "E" vi verrà proposta un'altra scelta: "Introduci 1 per sottrarre l'attuale tabella

di pagamento oppure 2 per avere il menù". Se introdurrete 1 il calcolatore sottrarrà l'ammontare totale delle spese ricorrenti che avete introdotto prima. Stamperà quindi il bilancio prima e dopo che le spese ricorrenti sono state sottratte.

Il programma tornerà quindi al menù di partenza.

A questo punto stamperà: "Il bilancio attuale è di Lit..." e un numero diverso da zero (almeno così spero) diversamente dalla prima volta che avete fatto girare il programma. Tutte le variabili sono salvate assieme al programma quando questo viene registrato su cassetta; ma iniziare il programma la volta successiva con un comando RUN o CLAER le azzererebbe.

L'ultima linea (GO TO 430) assicura che dopo la prima volta il programma parta automaticamente conservando il bilancio attuale.

Notate, quando fate girare questo programma, l'uso delle INK e della FLASH per evidenziare parte dei messaggi. Il colore e gli altri effetti grafici dovrebbero essere usati ovunque sia necessario ridurre al minimo le possibilità di errore da parte dell'operatore. Secondo questo principio anche le frasi di input dovrebbero essee "a prova di errore" dove è possibile, inmodo che un errore in fase di introduzione non provochi un disastro.

Per esempio la linea 325 rifiuta un input nullo della linea 320. Un input nullo (premendo solo ENTER senza aver precedentemente premuto un numero) avrebbe provocato un disastro alla linea 340. Analogamente, sebbene il programma chieda (alla linea 310) una "E" da introdurre quando si è terminata la memorizzazione delle entrate, il programma accetterà anche una "e" dato che non ci sono istruzioni nel programma che assicurino che il CAPS LOCK sia inserito.

```
  10 REM CONTABILITA' PERSONALE
  12 LET BALANCE=0
  15 GO TO 430
  20 PRINT '''"Introdurre il num
ero di spese che devono essere p
agate ogni mese"
  30 INPUT NUMBER: CLS
  40 DIM A$(NUMBER,12)
  50 DIM A(NUMBER)
  60 FOR A=1 TO NUMBER
  70 PRINT AT A-1,0;"Introdurre
il nome della spesa"; FLASH 1;A
  80 INPUT A$(A)
  90 PRINT AT A-1,0;"Quanto si s
pende mensilmente per ";A$(A);"
(in Lit)"
 100 INPUT A(A)
 110 PRINT AT A-1,0;A;TAB 3;A$(A
```

```
);TAB 15;"L";A(A);"
                                    "
 120 NEXT A
 130 PRINT '"Se e' tutto OK prem
i", FLASH 1;"ENTER"; FLASH 0;"Se
 c'e' un errore,","introduci il
numero del gruppo,","da cambiare
"
 140 INPUT B$
 150 IF B$="" THEN GO TO 260
 160 LET B=VAL B$
 170 PRINT "Introduci il nuovo n
ome";B
 180 INPUT A$(B)
 190 PRINT "Quanto si spende men
silmente per ";A$(B);" (in Lit)"
 200 INPUT A(B)
 210 CLS
 220 FOR A=1 TO NUMBER
 230 PRINT A;TAB 3;A$(A);TAB 15;
"L";A(A)
 240 NEXT A
 250 GO TO 130
 260 CLS
 265 IF BALANCE<>0 THEN GO TO 29
0
 270 PRINT ''"Introduci l'ultimo
 bilancio noto"
 280 INPUT BALANCE
 290 CLS
 300 PRINT AT 0,0;"Lit";BALANCE
 310 PRINT ''"Introduci le entra
te una per una includendo il sal
ario","termina le entrate introd
ucendo", FLASH 1;"E"
 320 INPUT Q$
 325 IF Q$="" THEN GO TO 320
 330 IF Q$="E" OR Q$="e" THEN GO
 TO 360
 340 LET BALANCE=BALANCE+VAL Q$
 350 GO TO 300
 360 INPUT "Introduci 1 per sott
rarre tutto dall'attuale tabella
```

```
 di pagamento oppure 2 per avere
 il menu'";C
 365 IF C=2 THEN GO TO 450
 370 LET SPEND=0: FOR A=1 TO NUM
BER
 380 LET SPEND=SPEND+A(A)
 390 NEXT A
 400 CLS
 410 PRINT ''"Il bilancio prima
delle          spese attuali era
 di Lit";BALANCE
 420 LET BALANCE=BALANCE-SPEND
 430 PRINT ''"Il bilancio attual
e e' di Lit";BALANCE
 440 PAUSE 200
 450 PRINT '"Introdurre"
 460 PRINT '"1 - Per inizializza
re i dati"
 470 PRINT '"2 - Per modificare
la tabella di pagamento"
 480 PRINT '"3 - Per depositare
denaro"
 485 PRINT '"4 - Per salvare su
nastro"
 490 PRINT '"5 - Per finire"
 500 INPUT C: CLS
 510 IF C=1 THEN LET BALANCE=0:
GO TO 20
 520 IF C=2 THEN GO TO 210
 530 IF C=3 THEN GO TO 300
 540 IF C=4 THEN SAVE "CONTABILI
TA"
 550 IF C=5 THEN STOP
 560 GO TO 430
```

AGENDA

Il programma Agenda può essere usato al posto di qualunque sistema a schede e, sebbene operi per giorni, può facilmente essere modificato in modo da operare per ore, se le vostre necessità sono tali.

Questo è un programma molto semplice rispetto agli altri del capitolo perchè i

suoi compiti possono essere schematizzati in questo modo: (a) accettare una data ed un appuntamento; (b) ordinare gli elementi cronologicamente; (c) stamparli a richiesta.

Comunque il programma non è semplice per il modo in cui gli daremo le date. Il programma accetterà solo date in questa forma: numero del giorno, una barra (/), numero del mese, una barra e l'anno. Basta riflettere un momento per capire che se il calcolatore ignorasse le sbarre e ordinasse i numeri così ottenuti, il 12 dicembre 1901 sarebbe considerato come successivo al 1 gennaio 1999 (in quanto 121201 è più grande di 10199).

Quindi il programma deve essere in grado di manipolare le date in modo da considerare come numeri più piccoli le date più vecchie, ma deve anche poterle stampare in una forma leggibile. Un altro problema viene dal fatto che un giorno può essere espresso con una o due cifre e lo stesso vale per il mese. Il calcolatore deve essere in grado di aggiungere uno zero davanti ai giorni e ai mesi ad una sola cifra (e naturalmente deve sapere quando c'è bisogno di uno zero).

La routine che fa tutto questo compare dalla linea 40 alla linea 80.

La linea 40 vi chiede di introdurre la data dando un esempio della forma richiesta. La linea 50 pone un'altra stringa (A$) uguale alla data, in modo che possa essere ristampata in seguito. La linea 60 controlla se il secondo elemento della stringa è una barra; così facendo viene a sapere se il giorno è a singola cifra e di conseguenza aggiunge uno zero all'inizio. La linea 70 controlla se il quinto elemento della stringa è una barra (e lo sarà solo se il mese è composto da un'unica cifra) e se è così aggiunge uno zero prima delle cifra che rappresenta il mese. Infine alla linea 80 la data viene cambiata in modo che compaia per primo l'anno, poi il mese e infine il giorno (la notazione americana delle date prevede che prima del giorno venga introdotto il mese; quindi alla linea 80 bisognerà cambiare 4 TO 5 con 1 TO 2 così come alla linea 40 sarà necessario cambiae 25/12/84 con 12/25/84).

La linea 90 stampa la data nella sua forma originale (ecco perchè abbiamo messo LET A$ = B$ ala linea 50) nell'angolo in basso a destra. Vi viene poi chiesto di inserire l'appuntamento relativo a quella data. Come nel programma precedente avete la possibilità di correggere un eventuale erore di battitura. Assumendo che non ci siano dati da modificare, il programma introduce la data (in una forma leggibile al computer) e l'appuntamento nella matrice D$ che contiene la vostra agenda.

La routine che inizia alla linea 220 ordina il contenuto dell'agenda e lo stampa in una forma comprensibile, la linea 330 provvede alla conversione della data. Come esercizio potreste aggiungere un menù, come nel programma precedente, per permettervi di salvare l'agenda su cassetta o di cancellarlo. Anche nella forma attuale se registrate l'agenda su cassetta o poi mandate in esecuzione il programma con GO TO 40 (invece che con RUN) manterrete il contenuto della agenda e sarete in grado di aggiungere altri elementi che verrebbero automaticamente ordinati.

```
10 REM AGENDA
20 DIM D$(200,32)
```

```
  30 FOR D=1 TO 200
  40 INPUT "Introduci la data ne
lla forma","25/12/84";B$
  50 LET A$=B$
  60 IF A$(2)="/" THEN LET A$="0
"+A$
  70 IF A$(5)="/" THEN LET A$=A$
( TO 3)+"0"+A$(4 TO )
  80 LET A$=A$(7 TO 8)+A$(4 TO 5
)+A$(1 TO 2)
  90 PRINT AT 0,0;"Data: ";B$
 100 PRINT '''"Introdurre l' appu
ntamento","(non piu' di 22 lette
re)"
 110 INPUT C$
 120 CLS
 130 PRINT AT 0,0;B$;"-";C$
 140 PRINT '''"Se e' tutto OK,pre
mi"; INVERSE 1;"ENTER"; INVERSE
0,"se qualcosa non va premi"; FL
ASH 1;"~E~"; FLASH 0;"e poi ENTE
R"
 150 INPUT E$: CLS
 160 IF E$<>"" THEN GO TO 40
 170 LET D$(D)=A$+C$
 180 PRINT '''"Premi"; INVERSE 1;
"ENTER"; INVERSE 0;"per introdur
re il prossimo      elemento,o u
n qualsiasi tasto  seguito da EN
TER per la stampa"
 190 INPUT E$: CLS
 200 IF E$="" THEN NEXT D
 210 PRINT PAPER 2; FLASH 1;"Att
endere prego....": PAUSE 100: CL
S
 220 LET B=0
 230 LET G=D
 240 LET Z=1
 250 LET B=Z+1
 260 IF B>G THEN GO TO 330
 270 IF D$(B)>D$(Z) THEN GO TO 2
90
 280 LET Z=Z+1: GO TO 250
```

```
290 LET Q$=D$(Z)
300 LET D$(Z)=D$(B)
310 LET D$(B)=Q$
320 GO TO 280
330 PRINT D$(G)(5 TO 6);"/";D$(
G)(3 TO 4);"/";D$(G)(1 TO 2);"-"
;D$(G)(7 TO )
340 LET G=G-1
350 IF G>0 THEN GO TO 240
```

RUBRICA

Dovreste essere in grado di stabilire un collegamento fra AGENDA e RUBRICA e il programma che segue: BASE DI DATI. Tutti e tre in sostanza accettano le informazioni dell'utente, le ordinano in qualche modo (o modi nel caso di BASE DI DATI) e quindi le stampano in ordine o ne permettono la ricerca. Le strutture dei tre programmi dovrebbero darvi sufficienti cognizioni per mettervi in grado di compilare un programma tipo archivio ordinato, tale da soddisfare le vostre necessità.

I menù della rubrica presenta queste possibilità:

Introduci il numero

1 — Per inizializzare una nuova rubrica
2 — Per aggiungere nuovi numeri
3 — Per cercare un numero conoscendo il nome
4 — Per salvare la rubrica
5 — Per stampare tutta la rubrica
6 — Per terminare

Nella sua forma attuale il programma accetterà fino a 200 elementi di massimo 32 caratteri, ma se siete in grado di usare solo nomi di battesimo potreste facilmente aumentare il numero massimo degli elementi cambiando la linea 20 con una DIM D$(400,16). Introducete il nome e il numero telefonico ad esso relativo e (come già fatto negli altri programmi) vi verrà data la possibilità di rivederlo ed eventualmente di modificarlo prima che sia aggiunto alla rubrica.

Per introdurre un altro nome, premete ENTER e il programma tornerà alla linea 40 per il prossimo nome della rubrica.

Se volete ordinare tutto l'elenco premete qualsiasi tasto (eccetto BREAK) e fatelo seguire da ENTER. La rubrica è ordinata alfabeticamente, partendo dalle prime let-

tere del nome (quindi dovete introdurre prima i cognomi se volete la rubrica ordinata sui cognomi). Il programma ritorna quindi al menù. Se introducete 3 "Cercare un numero conoscendo il nome" il programma andrà alla linea 500 e stamperà "Introduci il nome richiesto". Il computer ricerca nella rubrica (impiegando un tempo abbastanza breve) e stampa il numero desiderato (tenete presente che viene ricercato un nome identico a quello richiesto, spazi compresi) oppure stampa il messaggio "Nome non esistente" se il nome non è reperibile.

Questo programma come CONTABILITÀ PERSONALE vi permette di salvare la rubrica su cassetta e di farla caricare automaticamente mantenendo inalterato il suo contenuto. Se desiderate partire manualmente, senza perdere i dati, usato GO TO 360 invece di RUN.

```
  10 REM RUBRICA
  15 GO TO 360
  20 DIM D$(200,32)
  30 FOR D=1 TO 200
  40 INPUT "Introduci il nome ";
B$
  90 PRINT AT 0,0;"Nome: ";B$
 110 INPUT "Introduci il numero
di telefono ";C$
 120 CLS
 130 PRINT AT 0,0;B$;"  ";C$
 140 PRINT '';"Se e' OK premi ";
INVERSE 1;"ENTER"; INVERSE 0,"Se
 no premi "; FLASH 1;"~E~"; FLAS
H 0,"quindi ENTER"
 150 INPUT E$: CLS
 160 IF E$<>"" THEN GO TO 40
 170 LET D$(D)=B$+" "+C$
 180 PRINT '';"Premi "; INVERSE 1
;"ENTER"; INVERSE 0;"per introdu
rre il prossimo,","oppure un qua
lsiasi tasto ed ENTER","per ordi
nare l'elenco"
 190 INPUT E$: CLS
 200 IF E$="" THEN NEXT D
 210 PRINT PAPER 2; FLASH 1;"Sto
 ordinando..."
 215 POKE 23692,0
 220 LET B=0
 230 LET G=D
```

```
 240 LET Z=1
 250 LET B=Z+1
 260 IF B>G THEN GO TO 330
 270 IF D$(B)>D$(Z) THEN GO TO 2
90
 280 LET Z=Z+1: GO TO 250
 290 LET Q$=D$(Z)
 300 LET D$(Z)=D$(B)
 310 LET D$(B)=Q$
 320 GO TO 280
 330 PRINT D$(G)
 340 LET G=G-1
 350 IF G>0 THEN GO TO 240
 360 PRINT '''"Introduci il numer
o:"
 370 PRINT '"1 - Per inizializza
re una nuova   rubrica"
380 PRINT '"2 - Per aggiungere n
uovi numeri"
 390 PRINT '"3 - Per cercare un
numero conos-cendo il nome"
 400 PRINT '"4 - Per salvare la
rubrica"
 405 PRINT '"5 - Per stampare tu
tta la rubrica"
 410 PRINT '"6 - Per terminare"
 420 INPUT B: CLS
 430 IF B=1 THEN GO TO 20
 440 IF B=2 THEN NEXT D
 450 IF B=3 THEN GO TO 500
 460 IF B=4 THEN SAVE "RUBRICA"
 465 IF B=5 THEN FOR A=D TO 1 ST
EP -1: LPRINT D$(A): NEXT A
 470 IF B=6 THEN STOP
 480 GO TO 360
 500 PRINT '''"Introduci il nome
richiesto"
 510 INPUT A$: LET F=LEN A$
 520 PRINT '' FLASH 1; INK 1;"St
o cercando ";A$
 530 FOR A=1 TO D
 540 IF D$(A)( TO F)=A$ THEN PRI
NT ''D$(A)(F+1 TO ): GO TO 360
```

```
550 NEXT A
560 PRINT '"Nome non esistente"
570 GO TO 360
```

BASE DI DATI

Questo programma, che é in realtà una forma più elaborata di RUBRICA, é stato realizzato per gestire una collezione di dischi e per memorizzarli in due archivi, uno per i nomi degli artisti e l'altro per il titolo del disco.

È stato incluso in questo capitolo perchè può rappresentare molto bene un sistema di archivio aziendale. Come tutti i programmi guidati da menù, vengono presentate le varie possibilità.

Selezionare un'operazione.
1 — Creazione del nuovo Archivio.
2 — Inserimento dei nuovi elementi
3 — Stampa per nome
4 — Stampa per titolo
5 — Ricerca per nome
6 — Ricerca per titolo
7 — Memorizzazione su nastro
8 — Fine.

Il programma vi chiede di introdurre il NOME ARTISTA quindi il TITOLO e poi il CODICE. CODICE é il vostro sistema di riferimento. Il programma non ordina i dati usando il codice, ma semplicemente lo aggiunge alla fine.

La ricerca per titolo produrrà il titolo, l'artista e il codice e la ricerca per artista produrrà artista, titolo e codice. La matrice F$ contiene il file ordinato secondo l'artista; quello ordinato per titoli é contenuto nella matrice E$. Quando ricerca procedendo per nome o per titolo il programma confronterà solo le prime quattro lettere del nome.

```
  10 REM BASE DI DATI
  20 PRINT '''"Selezionare un'op
zione"
  25 PRINT '"1 - Creazione del n
uovo archivio"
  30 PRINT '"2 - Inserimento dei
 nuovi elementi"
  40 PRINT '"3 - Stampa per nome
"
```

```
  50 PRINT '"4 - Stampa per tito
lo"
  60 PRINT '"5 - Ricerca per nom
e"
  70 PRINT '"6 - Ricerca per tit
olo"
  80 PRINT '"7 - Memorizzazione
su nastro"
  90 PRINT '"8 - Fine"
 100 INPUT A
 110 CLS
 120 IF A=1 THEN GO SUB 210
 130 IF A=2 THEN NEXT J
 140 IF A=3 THEN GO SUB 630
 150 IF A=4 THEN GO SUB 670
 160 IF A=5 THEN GO SUB 710
 170 IF A=6 THEN GO SUB 800
 180 IF A=7 THEN SAVE "BASE"
 190 IF A=8 THEN STOP
 200 GO TO 20
 210 DIM F$(130,24): DIM E$(130,
24)
 220 POKE 23692,0
 230 FOR J=1 TO 130: CLS
 240 PRINT AT 0,0; INK 2;"Introd
uci "Z" per terminare","gli inse
rimenti"
 250 PRINT AT 10,10;"Elemento Nu
mero"; FLASH 1;J
 260 INPUT "Introduci il nome ar
tista ";A$
 270 IF A$="Z" THEN GO TO 390
 280 PRINT 'A$
 290 INPUT "Introduci il titolo
";T$
 300 PRINT 'T$
 310 INPUT "Introduci il codice
";C$
 320 PRINT 'C$
 330 PRINT '"Se e' OK premi "; I
NVERSE 1;"ENTER"; INVERSE 0,"Se
no premi un tasto qualsiasi e po
i ENTER"
```

```
 340 INPUT Z$
 350 IF Z$<>"" THEN GO TO 240
 360 LET F$(J)=A$+":"+T$+":"+C$
 370 LET E$(J)=T$+":"+A$+":"+C$
 380 NEXT J
 390 CLS
 400 PRINT INK 1; FLASH 1;"Sto o
rdinando..."
 405 LET G=J: LET B=0
 410 LET Z=1
 440 LET B=Z+1
 450 IF B>G THEN LET G=G-1: IF G
>0 THEN GO TO 410
 455 IF G=0 THEN GO TO 520
 460 IF F$(B)<F$(Z) THEN GO TO 4
80
 470 LET Z=Z+1: GO TO 440
 480 LET Q$=F$(Z)
 490 LET F$(Z)=F$(B)
 500 LET F$(B)=Q$
 510 GO TO 470
 520 LET G=J: LET B=0
 530 LET Z=1
 540 LET B=Z+1
 550 IF B>G THEN LET G=G-1: IF G
>0 THEN GO TO 530
 560 IF G=0 THEN CLS : GO TO 20
 570 IF E$(B)<E$(Z) THEN GO TO 5
90
 580 LET Z=Z+1: GO TO 540
 590 LET Q$=E$(Z)
 600 LET E$(Z)=E$(B)
 610 LET E$(B)=Q$
 620 GO TO 580
 630 FOR M=1 TO J
 640 LPRINT F$(M)
 650 NEXT M
 660 RETURN
 670 FOR M=1 TO J
 680 LPRINT E$(M)
 690 NEXT M
 700 RETURN
 710 PRINT "Introduci il nome de
```

```
ll'artista"
 720 INPUT N$: LET N$=N$+"    "
 730 FOR M=1 TO J
 740 IF F$(M)( TO 4)=N$( TO 4) T
HEN GO TO 770
 750 NEXT M
 760 PRINT "Nome non reperibile"
: PAUSE 200: RETURN
 770 PRINT F$(M)
 780 PAUSE 0
 790 RETURN
 800 PRINT "Inroduci il titolo"
 810 INPUT N$: LET N$=N$+"    "
 820 FOR M=1 TO J
 830 IF E$(M)( TO 4)=N$( TO 4) T
HEN GO TO 860
 840 NEXT M
 850 PRINT "Titolo non reperibil
e": PAUSE 200: RETURN
 860 PRINT E$(M)
 870 PAUSE 0
 880 RETURN
```

MODELLO FINANZIARIO / PROIEZIONI DI VENDITA

Come ho spiegato all'inizio di questo capitolo, il MODELLO FINANZIARIO é una versione primitiva di programmi come il Visicalc, che simulano un "foglio elettronico". Questo programma é stato progettato per accettare dati relativi alle vendite mensili, o alle quantità di articoli rifiutati, di cambiamenti nel personale o a qualunque altro evento che si presenta con una certa regolarità; da queste informazioni il programma estrapola previsioni sui mesi futuri, assumendo che tutti gli altri fattori rimangano inalterati.

Quando fate girare il programma, vi viene chiesto prima di tutto se volete un "hard copy" (copia su stampante) del risultato dell'elaborazione.

Se introducete 1 che per Si tutti i risultati importanti del programma, escluse le risposte dell'utente, verranno stampate esattamente come sullo schermo. Poi vi verrà chiesto "Per quanti mesi sono disponibili i dati".

Potete cambiare il mese con il giorno o con l'anno o con qualsiasi periodo di tempo su cui lavorate. Vi verrà poi chiesto di inserire i dati per ogni mese; al momento il programma é predisposto per ricevere dati che si riferiscono al massimo a 19 mesi. Cambiate l'"A-1" all'inizio della linea 100 con uno 0 se avete dati per più di 19 mesi.

A questo punto lo Spectrum elaborerà la variazione approssimativa da un mese all'altro. Confronterà cioè il secondo mese con il primo; il terzo con il secondo e cosi via.

La media percentuale della variazione sarà il risultato finale.

Sarete in grado di fare un'estrapolazione dei risultati specificando il numero di mesi su cui volete la proiezione. Se volete la proiezione può essere basata sull'ultimo mese piuttosto che sulla variazione media.

Una volta che ciò è stato fatto, il menù vi mostra come al solito le possibili scelte:

1 – Altro svolgimento della proiezione (il che significa che potete cambiare i dati dell'ultimo mese con la media o viceversa, e usare un periodo di tempo più lungo oppure più corto).

2 – Proiezione uguale alla precedente (anche se dovrete rispondere ancora alle domande del programma).

3 – Svolgimento ex-novo (questo cancella tutti i dati attuali).

4 – Memorizzazione su nastro (potrete recuperare dati tramite l'opzione 2 del menù ricaricandoli da cassetta).

5 – Fine.

Se avete fatto girare il programma senza usare la stampante e decidete di avere un "hard copy", usate la opzione 5 per fermare il programma quando raggiunge il menù; poi invece di far ripartire il programma con RUN dategli il comando GO TO 10. Selezionate l'opzione 2 del menù e avrete i risultati sulla stampante senza doverli reintrodurre.

```
  10 REM MODELLI FINANZIARI
        PROIEZIONI DI VENDITA
  15 GO SUB 490
  20 PRINT '''"Per quanti mesi so
no disponibili dati?"
  30 INPUT M: LET TOT=0
  40 IF M<2 THEN GO TO 30
  50 CLS : DIM A(M): DIM B(M)
  60 POKE 23692,0
  70 FOR A=1 TO M
  80 PRINT AT 20,0;"Introduci i
dati del mese "; FLASH 1; INK 2;
A
  90 INPUT A(A)
 100 PRINT AT A-1,0; BRIGHT 1; P
```

```
APER 1; INK 7;"  Mese  ";A;" ";T
AB (17-LEN STR$ A(A));A(A);"
   "
 105 IF Z=1 THEN LPRINT     "  Me
se  ";A;" ";TAB (17-LEN STR$ A(A
));A(A);"       "
 110 LET TOT=TOT+A(A): NEXT A
 115 PRINT AT 20,0;"
                    ";AT A,0;
 120 LET AV=TOT/M: FOR B=2 TO M
 130 LET B(B)=(100-(A(B-1)*100/A
(B)))
 140 NEXT B
 145 PAUSE 100
 150 PRINT ''' FLASH 1; INK 1;"D
ifferenza tra i mesi?"
 155 IF Z=1 THEN LPRINT "Differe
nza tra i mesi?"
 160 FOR A=2 TO M
 170 PRINT "Dal mese  "; INK 1;A
-1, INK 0;TAB 5;"Al mese "; INK
1;A; INK 0;" - ";INT (B(A)+.5);"
%"
 175 IF Z=1 THEN LPRINT "Dal mes
e  ";A-1;"AL mese ";A;" - ";INT
(B(A)+.5);"%"
 180 NEXT A
 185 PAUSE 100
 190 LET TOTAL=0
 200 FOR A=2 TO M
 210 LET TOTAL=TOTAL+B(A)
 220 NEXT A
 230 LET AVERAGE=INT (TOTAL*100/
(M-1))/100
 235 PRINT ' INK 2;"-------------
-------------------------"
 236 IF Z=1 THEN LPRINT "---------
-------------------------------"
 240 PRINT '"Variazione media";
INK 1; FLASH 1;AVERAGE; INK 0; F
LASH 0;"%"
 245 IF Z=1 THEN LPRINT "Variazi
one media";AVERAGE;"%"
```

```
 250 PAUSE 100
 255 PRINT ' INK 2;"----------
----------"
 256 IF Z=1 THEN LPRINT "-----
----------"
 260 PRINT '"Ecco una proiezione
 di variazioni"
 265 IF Z=1 THEN LPRINT "Ecco un
a proiezione di variazioni"
 270 INPUT "Quante ne vuoi?";NUM
BER
 280 PRINT '"L'ultimo mese regis
trato era ";A(M)
 290 PRINT '"La media mensile er
a ";10*(INT AV/10)
 300 INPUT "Vuoi che la proiezio
ne si basi  su 1 - mese finale;o
               2 - media mensile?
";D
 305 PRINT ' INK 2;"----------
----------"
 306 IF Z=1 THEN LPRINT "-----
----------"
 310 LET E=(A(M) AND D=1)+((10*(
INT AV/10)) AND D=2)
 315 PRINT '"Mese 1,registrato:"
;A(M)
 316 IF Z=1 THEN LPRINT "Mese 1,
registrato:";A(M)
 320 FOR A=2 TO NUMBER
 325 POKE 23692,0
 330 LET E=E+AVERAGE*E/100
 340 PRINT "Mese ";A;",proiettat
o: ";INT (E)
 342 IF Z=1 THEN LPRINT "Mese ";
A;",proiettato: ";INT (E)
 345 NEXT A
 350 PAUSE 100: POKE 23692,0
 355 PRINT ' INK 2;"----------
----------"
 356 IF Z=1 THEN LPRINT "-----
----------"
 360 PRINT ' INK 2; INVERSE 1;"S
```

```
cegli tra:"
 370 PRINT TAB 3;"1 - Altro svol
gimento della proie-  zione"
 380 PRINT TAB 3;"2 - Proiezione
 uguale alla prece-   dente"
 390 PRINT TAB 3;"3 - Svolgiment
o ex-novo"
 400 PRINT TAB 3;"4 - Memorizzaz
ione su nastro"
 405 PRINT TAB 3;"5 - Fine"
 410 LET A$=INKEY$
 420 IF A$="" THEN GO TO 410
 430 IF A$="1" THEN GO TO 260
 440 IF A$="2" THEN GO TO 150
 450 IF A$="3" THEN RUN
 460 IF A$="4" THEN SAVE "VENDIT
E"
 470 IF A$="5" THEN STOP
 480 GO TO 360
 490 INPUT "Vuoi l'hard copy?
                          1 - Si
                          2 - No";
Z
 500 IF Z<1 OR Z>2 THEN GO TO 49
0
 510 RETURN
```

BIBLIOGRAFIA

- *"Business Information Processing With BASIC"* Struble George (Addison-Wesley Publishing Company, USA 1980).
- *"BASIC Computer Programs For Business"*. Stenberg Charles D. (Hayden Book Company, USA 1981).
- *"BASIC Computer Programs In Scienze And Engineering"*. Gilder Jules H. (Hayden Book Company, USA 1981).
- *"Some Common BASIC Programs"*. Poole, Lou and Borchers, Mary.

CAPITOLO 5

USARE LO SPECTRUM A SCUOLA

Questo capitolo é stato scritto per coloro che vogliono usare lo Spectrum come uno strumento per insegnare, come i genitori che desiderano aiutare i figli o gli insegnanti che intendono usare il calcolatore a scuola. Comunque non c'è ragione perchè non possiate usare questi programmi e le loro idee, se volete, per ripassare in vista di possibili esami, o anche solo come piccolo esercizio mentale. È interessante notare quanto potete imparare solo scrivendo un programma che aiuti gli altri ad apprendere. Come minimo potrete imparare un po' di francese o di fisica o di qualsiasi altra cosa, solo guardando le informazioni da memorizzare nelle frasi DATA per un programma-quiz.

Il software educativo per lo Spectrum o per altri computer analoghi che si trova in commercio tende a soffrire di due grosse mancanze. O é troppo semplice e generale, tanto da risultare inutile per chiunque oppure é talmente specializzato che è valido solo per pochissime persone.

Creare il vostro proprio "strumento di conoscenza" col computer sopperisce ad entrambe le mancanze. Potete adeguare i programmi adattandoli alle vostre necessità, scrivendo e provando solo quelle parti che vi sembrano interessanti. Le routine ed i programmi discussi in questo capitolo possono essere facilmente adattati ai vostri specifici bisogni.

Il capitolo si svilupperà su programmi che permettono di fare pratica su conoscenze fondamentali come l'aritmetica e (nel caso di un programma) la conoscenza del francese. Ci sono molti altri tipi di programmi educativi che potete scrivere per insegnare anche in altri campi, oppure per risparmiare tempo in laboratorio (ad esempio un programma che combina gli elementi chimici con le relative quantità ben specificate e che sia in grado di mostrare i composti risultanti). Nella maggior parte dei casi troverete che i programmi qui illustrati vi serviranno come punto di partenza per produrre i programmi di cui avete bisogno.

I programmi educativi, come vedremo in seguito, non devono essere noiosi. Qualche programma educativo può anche essere scritto esplicitamente come fosse un gioco.

Questo tipo di programma richiede che i giocatori sviluppino e applichino la loro conoscenza per poter competere con altri studenti. Per fare un buon lavoro bisogna fare in modo che il successo vada a chi utilizza il più efficacemente possibile la pro-

pria conoscenza e, sebbene fattori casuali possano avere qualche peso, non dovrebbero determinare il risultato della gara.

I programmi che insegnano nuovi concetti potrebbero essere molto lunghi, perché il programmatore deve anticipare tutti gli errori e le cattive interpretazioni che questi programmi possono sollevare e quindi fornirli di sequenze di correzione adeguate.

Lo stesso concetto dovrà essere presentato in parecchi modi differenti, in modo che un bambino che non abbia capito la prima spiegazione, possa arrivarci quando l'informazione è presentata in un modo diverso.

Ciò potrebbe sembrare eccessivo, quasi impossibile; una via di mezzo potrebbe essere un programma scritto con il presupposto che lo studente abbia con sè il libro di testo. In questo caso il programma é usato come una guida intelligente al libro piuttosto che come un sostituto dello stesso. Il programma ad esempio potrebbe dire: «Adesso che hai mostrato di aver capito che le credenze e i culti del Lamaismo derivano dalla forma Mahaiana del Buddismo vorrei che tu andassi a pagina 26 del testo e leggessi come il Lamaismo fu introdotto in Tibet. Quando l'avrai fatto, torna da me, per qualche domanda in proposito». Al ritorno le domande potrebbero controllare i concetti esposti a pag. 26 e, se qualche lacuna é stata evidenziata, invitare lo studente a portarsi ad una specifica posizione sulla pagina, o ad una qualsiasi altra parte del libro, dove si faccia riferimento a quell'argomento. Usare un programma in questo modo assicura che il tempo del programmatore non é sprecato per creare innumerevoli frasi PRINT per coprire ogni possibilità e questo significa che il vostro Spectrum può essere usato per aiutare l'insegnamento di materie che sembrerebbe invece troppo complesse per permettere una loro computerizzazione.

Le ricerche sono un mezzo abituale nella scuola elementare, ma gli studenti incontrano spesso dei problemi nel procurarsi le informazioni relative. Lo Spectrum può essere usato per memorizzare le informazioni, questo può essere ottenuto usando un menù di approccio tipo il Televideo; o può essere usato per aiutare a collegarsi con un'adeguata fonte di informazioni.

Tralasciando gli immediati benefici che ciò potrebbe avere sugli studenti, questo modo di agire potrebbe aiutarli a rendersi conto dell'aumentata importanza della tecnologia informativa nella società moderna.

Se una scuola, o magari casa vostra, non ha la possibilità di collegarsi col Televideo, una simulazione di esso con un numero limitato di scelte può essere scritto per aiutare gli studenti a famigliarizzare con l'uso del menù come strumento per reperire informazioni memorizzate in una «struttura ad albero» indicizzato.

Altri tipi di simulazione vi permettono di portare a termine indagini che non possono essere eseguite nella vita reale. Lo Spectrum può usare un modello matematico per predire le conseguenze di certi cambiamenti apportati al sistema. Potreste ad esempio voler conoscere quanto l'inquinamento possa influenzare la vita dei pesci, degli insetti e delle piante in un lago.

Una simulazione nel settore economico vi potrebbe mettere in grado di alterare le riserve finanziarie e vedere come ciò si rifletterà sui livelli occupazionali. Una simu-

lazione chimica potrebbe permettervi di cambiare la catalisi, la temperatura e la pressione per ottimizzare ad esempio la resa del biossido di zolfo in un processo per la fabbricazione dell'acido solforico.

Naturalmente una simulazione può essere realistica solo quando il modello matematico che rappresenta il sistema é accurato.

Anche se il modello non é molto accurato, non appena gli studenti vengono a conoscenza delle semplificazioni imposte nella simulazione, questa teoria può risultare molto istruttiva.

Alla fine di questo capitolo discuterò alcuni approcci specifici al software educativo che sono stati provati con successo da altre persone, cercando di darvi un'idea sul tipo di cose che possono essere facilmente trasformate in programmi educativi.

Prima di tutto, vorrei discutere dei programmi specifici (alcuni scritti da Jeff Warren, e altri da me) che mostreranno diversi modi per avvicinarsi alla produzione di software educativo. Sebbene questi programmi potrebbero non essere di uso immediato per voi, almeno in questa forma, dovrebbero comunque darvi un modello da cui partire per produrre i vostri programmi.

Molti programmi educativi usano la funzione RND (generazione di numeri casuali) per predisporre somme o per selezionare domande a caso in modo che lo studente non si annoi con sequenze prevedibili. Il nostro primo programma genera casualmente semplici divisioni, ma può essere facilmente modificato per fare pratica con qualunque soggetto aritmetico fondamentale.

La linea 5 trasforma la lettera D in un simbolo grafico, simile al normale segno di divisione che si è soliti usare a scuola. Successivamente il punteggio iniziale, s, é posto uguale a zero. Il programma presenta poi 20 domande usando un ciclo FOR/NEXT in cui q è il numero della domanda (linee dalla 20 alla 120). La domanda prende questa forma "Quanto fa 32 diviso 4?"; lo Spectrum pone il risultato nella forma x/y. La linea 30 produce un valore di y tra 1 e 12, come la risposta finale.

La linea 50 presenta la domanda e accetta la risposta usando INPUT a. La linea 60 sceglie tra la sequenza di correzioni delle linee 70 e 80 e la frase di elogio alla linea 90, che inoltre aggiorna il punteggio. Ricordatevi che dovete premere assieme CAPS SHIFT e il tasto 9 nello stesso tempo per ottenere il cursore grafico G, prima di premere la D alla linea 80. La D cambierà quando farete girare il programma.

La linea 100 mette in grado l'utente di continuare con la prossima domanda solo quando è pronto.

Per quanto velocemente uno studente possa seguire lo sviluppo del programma, l'informazione sullo schermo varia incessantemente; così un ritardo prefissato non sarebbe una buona idea. Ad esempio potrebbe essere usata l'istruzione PAUSE, ma dovreste fissare un tempo piuttosto lungo, o il messaggio scomparirà prima di essere letto e una volta cancellato lo studente lo avrebbe definitivamente perso. Non forzate mai l'utente ad aspettare la parte successiva del programma usando dei cicli FOR/NEXT per creare ritardi fissi, perchè questo può essere molto frustrante. Notate che la linea 100 vi permette di fermare prematuramente il programma premendo "s" prima di ENTER. L'ultima linea del programma stampa il risultato finale.

```
   5 FOR f=0 TO 7: READ a: POKE
USR "d"+f,a: NEXT f: DATA 0,16,0
,254,0,16,0,0
  10 LET s=0
  20 FOR q=1 TO 20
  30 LET y=2+INT (RND*11)
  40 LET x=y*INT (RND*12+1)
  50 PRINT AT 9,0; PAPER 6; "Dom
anda ";q'': PRINT "Quanto fa ";x
;" diviso ";y;"?": INPUT a
  60 IF a=x/y THEN GO TO 90
  70 PRINT PAPER 6;''"Sbagliato.
 La risposta non e' ";a
  80 PRINT ''x;" D ";y;" = ";x/y
: PRINT PAPER 6;'"( ricorda che
";y;" x ";x/y;" = ";x;" )": GO T
O 100
  90 PRINT ''"Giusto, ";a;" e' e
satto.": LET s=s+1
 100 INPUT "Premi ENTER per cont
inuare.";c$: IF c$="s" THEN STOP
 110 CLS
 120 NEXT q
 130 PRINT AT 0,4;"Hai totalizza
to ";s;" su 20"
```

Assicuratevi di limitare la difficoltà delle domande in rapporto alla capacità dello studente. Non permettete allo Spectrum di porre domande del tipo "Quanto fa 19 diviso 7?" se lo studente non sa niente di rapporti decimali.

Potete anche rendere il vostro programma più adatto all'utente usando il punteggio per controllare la difficoltà della domanda. Un modo di ottenere questo nel programma delle divisioni è quello di riscrivere la linea 30 come:

```
30 LET y=2+INT (RND*(3+(s>=5)*4+(s>=10*4))
```

Finchè il punteggio che rappresenta le risposte giuste, si mantiene minore di 5, y sarà disponibile nella fascia da 2 a 4, perchè sia l'espressione logica (s>=5) che quella (s>=10) saranno uguali a zero (infatti come espressioni logiche saranno valutate false). Quando il punteggio arriva a valori maggiori o uguali a 5 la prima espressione logica diventa vera e quindi assume il valore 1; a y verranno dati di conseguenza valori tra 2 e 8. Infine quando il punteggio raggiunge 10, a y verrà dato al massimo un valore uguale a 12.

Allo stesso modo potete cambiare la linea 40 con:

```
40 LET x=y*INT (RND*(5+(s>=5)*7)+1)
```

Il vantaggio di questo tipo di controllo sul programma é che il passaggio a domande più difficili é ben dosato e l'esercizio sarà più facile e più piacevole per l'utente.

Sebbene il programma sia stato scritto per le divisioni, non é difficile cambiare l'argomento delle domande. Possiamo cambiare la seconda parte della linea 50 con:

```
"Quant'é 1/";y;" di";x;"?"
```

Adesso questa istruzione pone una domanda del tipo "Quant' é 1/4 di 32?" ed é chiaramente fatta per impratichire sulle frazioni. Avrete bisogno di cambiare la linea 80 per fronteggiare il cambiamento della domanda alla linea 50. Provate a scrivere altre versioni del programma per risolvere addizioni, sottrazioni (ricordatevi di fare il primo numero più grande del secondo se gli studenti non conoscono ancora i numeri negativi) e le moltiplicazioni.

A molti programmatori piace rendere più personale il proprio software introducendo nelle risposte del computer il nome dell'utente. L'idea é quella di fare apparire la macchina più umana. Se volete ottenere questo dovrete scrivere un programma che lavori così:

Spectrum: "Ciao! Come ti chiami?"
Studentessa: "Samantha"
Spectrum: "Bene Samantha, spero che tu sia pronta a rispondere ad alcune domande sull'anatomia del topo".

Si può proseguire con una lista di "frasi di incoraggiamento", una di queste è selezionata a caso per ogni risposta esatta:

"Giusto Samantha, tibia è esatto"

oppure

"Ben detto Samantha, femore è giusto"

Questo tipo di approccio dà molta soddisfazione a qualche bambino, specialmente a quelli che impazziscono per i computer e i robot parlanti della fantascienza.

Lo Spectrum può anche usare gli istinti competitivi dei bambini mantenendo memorizzato il punteggio record associato al miglior tempo, se utilizzate la variabile di sistema FRAMES per prendere il tempo delle risposte. Questo metodo risulta utile se avete uno studente che cerca continuamente di migliorare i propri risultati, ma può risultare scoraggiante per il bambino più lento del gruppo; quindi state attenti quando adottate questo modo di agire.

Il prossimo programma, QUIZ ARITMETICO mostra come può essere usato il nome dello studente e, in generale, ha un approccio più amichevole di quello del programma precedente. Inoltre lo studente può scegliere il tipo di domanda (addizione sottrazione moltiplicazione o divisione) che gli verrà posta.

```
  10 REM QUIZ ARITMETICO
  30 PRINT '''"Ciao! Adesso facc
iamo un po' di  calcoli."
  40 PAUSE 200
  50 CLS
  60 PRINT '''"Bene, come ti chi
ami?"
  70 INPUT A$
  80 CLS
  90 PRINT '''"Felice di incontr
arti"; FLASH 1; BRIGHT 1; INK 1;
A$
 100 PRINT '''"Premi un qualsias
i tasto quando sei pronto per in
cominciare."
 110 PAUSE 0: LET SCORE=0
 120 CLS
130 PRINT '"Bene, ";A$;", io pos
so proporti delle domande su:"
 140 PRINT '" 1 - Addizioni;"
 150 PRINT '" 2 - Sottrazioni;"
 160 PRINT '" 3 - Moltiplicazion
i; oppure"
 170 PRINT '" 4 - Divisioni."
 180 PRINT '"Introduci un numer
o da "; FLASH 1;"1 a 4"' FLASH 0
;"per dirmi quale tipo di calcol
i vuoi provare."
 190 INPUT B$: LET B=VAL B$
 200 IF B<1 OR B>4 THEN GO TO 19
0
 205 CLS
 210 PRINT '" Ok, ";A$;", farem
o alcuni calcoli usando ";
 220 PRINT INK 2; FLASH 1;("addi
zioni" AND B=1)+("sottrazioni"
AND B=2)+("moltiplicazioni" AND
```

```
B=3)+("divisioni" AND B=4)
 230 PAUSE 300
 240 CLS
 250 PRINT '''"Introduci 1 se vu
oi calcoli semplici"'" o 5 se li
 vuoi piu' complessi."'" (puoi i
ntrodurre 2,3 o 4 se vuoi domand
e ne' troppo facili ne' troppo d
ifficili.)"
 260 INPUT C$: LET C=VAL C$
 270 IF C<1 OR C>5 THEN GO TO 26
0
 275 CLS
 280 LET D$=("+" AND B=1)+("-" A
ND B=2)+("*" AND B=3)+("/" AND B
=4)
 290 FOR E=1 TO 10
 300 LET A=INT (RND*(10+C))+C
 310 LET B=INT (RND*(10+4*C))+C
 320 LET G$=STR$ A+D$+STR$ B
 325 IF ABS (INT VAL G$)<>VAL G$
THEN GO TO 300
 330 PRINT "Bene, ";A$;", questa
e' la domando numero ";E
 340 PRINT '''"Quanto fa ";G$;"?
"
 350 PAUSE 50
 360 PRINT '''"Ora prova, e intr
oduci la tua risposta."
 370 INPUT F
 375 PRINT ''
 380 IF F=VAL G$ THEN BEEP .08,F
+SCORE: BEEP 1,2*(F+SCORE): PRIN
T FLASH 1; INK RND*7; PAPER 9;"B
ene, ";A$: LET SCORE=SCORE+1
 390 IF F<>VAL G$ THEN PRINT "No
 spiacente, ";A$;", ma "'"La ris
posta e' ";VAL G$
 400 PAUSE 50
 410 PRINT '''"Ora il tuo punteg
gio e' "; INK 1; FLASH 1;SCORE;
INK 0; FLASH 0;" su ";E
 420 IF E<10 THEN PRINT '''"Prem
```

```
i un qualsiasi tasto per la pros
sima domanda.": PAUSE 0: CLS : N
EXT E
 430 PAUSE 100
 440 CLS
 450 PRINT '''"Il test e' finito
";A$;"."
 460 PAUSE 100
 465 PRINT '''"Il tuo punteggio
e' "; INK 1; FLASH 1;SCORE; INK
0; FLASH 0;" su ";E;TAB 8; FLASH
 1; BRIGHT 1; INK 2; SCORE*100/E
;"%"
 470 PRINT '''"Vorresti fare un'
altra partita?"'"Batti s se si,
o qualsiasi altro tasto per fini
re."
 480 INPUT Y$
 490 IF Y$="S" OR Y$="s" THEN GO
 TO 100
 500 PRINT '''"È' stato bello fa
re calcoli con te ";A$
 520 PRINT '' FLASH 1;"ARRIVEDER
CI"
 530 STOP
```

Come vedete, il colore e la FLASH sono stati usati per evidenziare certe cose (come il nome dello studente la prima volta che è usato). La conversazione del programma procederà in questo modo:

"Ciao! Adesso facciamo un po' di calcoli"
"Bene, come ti chiami?" (Lo studente introdurrà il suo nome)
"Felice di incontrarti (FLASH, BRIGHT, INK blu), Samantha"
"premi un qualsiasi tasto quando sei pronto per incominciare".

PAUSE 0 è usata qui (alla linea 110) per mantenere il testo finchè non viene premuto un tasto. La variabile SCORE (che tiene conto del punteggio) è posta uguale a zero, infine lo schermo viene pulito (linea 120). Poi il computer riprende la conversazione:

"Bene Samantha, io posso proporti delle domande su:
1 – Addizioni;
2 – Sottrazioni;
3 – Moltiplicazioni; oppure
4 – Divisioni

Introduci un numero da (FLASH inserito) 1 a 4 (FLASH disinserito) per dirmi quale tipo di calcoli vuoi provare".

Lo studente introduce un numero che come potete vedere, alla linea 190 è accettato come una stringa e quindi convertito (usando la VAL) in un numero.

È importante scrivere programmi in maniera tale che un input scorretto non determini un disastro. Ciò non viene totalmente sviluppato nel primo programma di questo capitolo, quello sulle divisioni, dove l'input di una lettera al posto di un numero causerebbe l'arresto del programma; verrebbe inoltre stampato un messaggio di errore che lascerebbe l'utente incerto sul da farsi. Questo problema può essere risolto trattando ogni input come una stringa.

Potete anche usare la funzione CODE per assicurarvi che l'input sia numerico, se è questo che vi serve. Quindi potete convertire la stringa in un numero usando la VAL.

Provate le linee seguenti per l'input un numero ad una sola cifra:

```
10 INPUT n$
20 IF LEN n$ <>1 THEN PRINT "Una sola cifra per favore": GO TO 10
30 IF CODE n$ >48 OR CODE n$ <57 THEN PRINT "Solo numeri per favore": GO TO 10
40 LET n=VAL n$: PRINT "Finalmente!"
```

Provatelo per vedere che potete terminare il programma solo introducendo un numero con una singola cifra. I numeri hanno codici da 48 a 57 compresi.

Per numeri a più cifre dovrete separare gli input in caratteri singoli usando tecniche di frazionamento delle stringhe e controllando ogni singolo carattere. Come alternativa potreste anche usare la INKEY$= per ottenere numeri di una sola cifra, code INKEY$ = 13 (che è il codice di carattere per ENTER) termina gli input.

Ora torniamo al nostro QUIZ ARITMETICO. Il calcolatore ha chiesto all'utente di introdurre un numero da 1 a 4 per selezionare il tipo di domanda che verrà posta. La conversazione continua:

"OK Samantha, faremo alcuni calcoli usando sottrazioni" (oppure qualsiasi altro tipo di operazione che Samantha ha scelto)

Notate come la linea 220 usi il modo dello Spectrum di valutare l'espressione logica AND per scegliere una delle quattro parole (addizione, sottrazione, moltiplicazio-

ne, divisione) a seconda del valore che è stato dato a B. C'è una PAUSE (linea 230) e lo schermo è cancellato. Il loquace Spectrum continua:

"Introduci 1 se vuoi calcoli semplici o 5 se li vuoi più complessi. (Puoi anche introdurre 2, 3 o 4 se vuoi domande nè troppo facili nè troppo difficili)".

Questo metodo dà all'utente un mezzo per controllare il programma; inoltre la natura interattiva dell'esperienza dovrebbe aiutarlo a sentirsi realmente in conversazione con un "robot intelligente" dotato di personalità.

L'input è accettato come una stringa e convertito in un numero, poi viene pulito lo schermo.

La linea 280 fa ancora uso della valutazione logica per determinare il segno aritmetico (+, —, * oppure /) che verrà usato nella presentazione delle domande all'utente.

La linea 290 fa partire il ciclo di 10 domande che verranno poste allo studente, con le linee 300 e 310 vengono scelti i numeri per il problema; la fascia dei numeri è in relazione col grado di difficoltà (variabile C) richiesto allo studente alla linea 260.

La linea 320 unisce i numeri e l'operatore aritmetico prescelto in una singola stringa. La linea 325 controlla il risultato del problema per assicurarsi che non sia una frazione (come avrebbe potuto essere in caso di una divisione) e neppure un numero negativo (come avrebbe potuto essere in una sottrazione). Questa linea di controllo potrebbe essere tralasciata se i numeri non interi e quelli negativi sono accettabili.

La linea 330 stampa la domanda usando ancora il nome dello studente.

"Bene Samantha, questa è la domanda numero 4"
"Quanto fa 7 + 3?"

A questo punto c'è una leggera pausa (linea 350) quindi lo Spectrum stampa:

"Ora prova e introduci la tua risposta"

La linea 380 controlla la risposta e se è corretta avvisa lo studente con suono; la cui intensità è correlata al numero di risposte esatte che lo studente ha dato nel corso del test. Le parole "Ben detto Samantha" sono stampate sullo schermo, in un colore lampeggiante scelto a caso (usando INK 9 per assicurarsi che le parole vengano lette) e il punteggio (SCORE) è incrementato di uno. Se la risposta data è sbagliata la linea 390 stampa:

"No spiacente Samantha ma la risposta è 10"

. Notate che non ci sono nè parole lampeggianti nè suoni, per una risposta sbagliata, perchè la risposta corretta merita un miglior "riconoscimento" di quella errata.

C'è un'altra piccola sosta (PAUSE) quindi lo Spectrum stampa:

"Il tuo punteggio è (INK 1, FLASH) 1 (FLASH disinserito, INK normale) su 3".

Se la decima domanda non è ancora stata posta, lo Spectrum continua:

"Premi un tasto qualsiasi per la prossima domanda"

Questa attesa per un tasto premuto permette allo studente di procedere al passo con cui si sente a suo agio.

Alla fine della decima domanda, dopo una PAUSE, lo schermo è cancellato.

"Il test è finito Samantha". (PAUSE)
"Il tuo punteggio è (INK 1, FLASH inserito) 8 (INK normale, FLASH disinserito) su 10. (Altri effetti) 80%"
"Vorresti fare un'altra partita? Batti 'S' se si, o qualsiasi altro tasto per finire".

Sebbene si possa impiegare molto tempo per scrivere un programma conversazionale come questo, l'aumentata interazione che esso crea con lo studente ripaga abbondantemente del tempo perso per creare le linee in più del programma.

La linea 480 accetta la risposta dello studente, la linea 490 controlla se è una 'S' o una 's' eliminando il problema del CAPS LOCK inserito o meno.

Se lo studente richiede un altro test il computer non ricomincia dall'inizio, in quanto così facendo cancellerebbe il nome dello studente dalla memoria e ripeterebbe la sequenza che chiede il nome allo studente; tutto questo distruggerebbe l'illusione di un robot intelligente.

Invece il computer va al punto del programma (linea 100) che segue la sequenza di richiesta del nome. Se lo studente non ha risposto 'S' o 's' alla domanda riguardante un ulteriore test, il calcolatore stampa: "È stato bello fare calcoli con te Samantha. (FLASH inserito). ARRIVEDERCI".

Sebbene la spiegazione sia stata più lunga del programma, spero che ciò mostri quanto efficace possa essere un programma anche semplice che faccia uso della "conversazione". Confrontando gli output di QUIZ ARITMETICO col programma delle divisioni si nota quanto più amichevole sia con l'utente il secondo rispetto al primo.

Ci sono tanti modi di rendere i programmi amichevoli verso l'utente, usando una retroazione maggiore di semplici messaggi tipo: "Ben detto". Molti bambini accolgono favorevolmente l'aggiunta motivazione che deriva dal collegare l'esercizio scolastico con qualche tipo di gioco o di visualizzazione interessante. Ora estenderemo il primo programma del capitolo in questo modo, facendo uso della grafica, del colore e del suono dello Spectrum.

In cima allo schermo il programma disegnerà un fiume con un ponte e un carro armato sulla riva sinistra. Ogni volta che c'è una risposta giusta la superficie del

ponte è estesa e se tutte e 20 le risposte saranno giuste verrà prodotta una musichetta di "vittoria", il carro armato passerà il ponte ed esploderà i suoi colpi.

Prima sostituite la linea 5 del programma delle divisioni con:

```
5 GO TO 150
```

quindi aggiungete alla fine della linea 90

```
... : GO SUB 250
```

cambiate la linea 110 con la seguente:

```
110 PRINT AT 8,0; : FOR n=1 TO 11 : PRINT b$: NEXT n
```

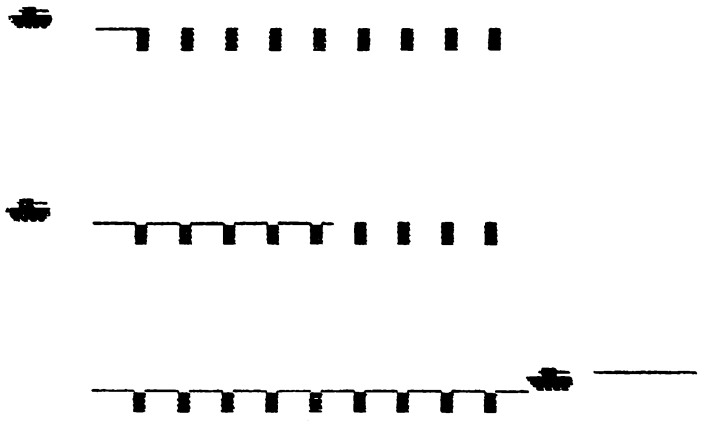
Poi aggiungete le seguenti linee al vostro programma.

```
 140 IF s=20 THEN PAUSE 75: GO T
O 260
 145 STOP
 150 FOR n=1 TO 4: READ s$
 160 FOR m=0 TO 7: READ a: POKE
USR s$+m,a: NEXT m
 170 NEXT n
 180 DATA "a",7,15,7,255,255,127
,63,27
 190 DATA "b",192,254,192,255,25
5,255,254,108
 200 DATA "c",255,0,0,0,0,0,0,0
 210 DATA "d",0,16,0,254,0,16,0,
0
 220 LET b$="
                "
 230 PRINT AT 3,0;"AB"': PRINT P
APER 4;"    ": PRINT AT 4,5;" ■
■ ■ ■ ■ ■ ■ ■ ■": PRINT AT 4,24
; PAPER 4;"          ": PRINT PAPE
R 1;b$;b$
 240 GO TO 10
 250 PRINT AT 4,3+s; OVER 1;"C":
 RETURN
 260 FOR n=0 TO 10: FOR m=4 TO 8
 STEP .5: BEEP 0.01,m*m: NEXT m:
```

```
 NEXT n
 270 FOR n=0 TO 23: PRINT AT 3,n
;" AB": PAUSE 5: NEXT n
 280 FOR n=1 TO 50: PLOT 216,150
: DRAW INK 2;36,0: PLOT 216,150:
 DRAW INK 7;36,0: NEXT n
 290 PRINT AT 9,10;"Fine."
```

Le linee dalla 150 alla 170 creano simboli grafici definiti dall'utente usando i dati contenuti nelle linee dalla 180 alla 210. Il b$ definito alla linea 220 è usato alla linea 110 per pulire solo una parte dello schermo lasciando le figure del ponte e del carro armato inalterate. Questa figura è creata alla linea 230. Dovete ottenere il cursore grafico prima di battere "AB" dato che questo sarà il carro, lo stesso vale per il "C" alla linea 250 e per "AB" alla 270.

La linea 250 disegna una nuova sezione della superficie del ponte dopo ogni risposta esatta. La linea 260 suona la musica di vittoria se tutte le risposte sono giuste. La 270 muove il carro sul ponte e la 280 lo fa sparare.



Potete, ovviamente, aggiungere la sequenza del carro armato a qualsiasi programma educativo, per fornirlo di questa ulteriore motivazione. Comunque, se usate troppo spesso questo giochetto, rischiate di fargli perdere il suo fascino. Provate a

fare altre sequenze del genere per vostro conto, tenendo presente i seguenti punti. Assicuratevi che la sezione di complimenti non interferisca con gli aspetti educativi del programma. Se facesse perdere più di un secondo dopo ogni domanda, diventerebbe ben presto noiosa. L'incentivo dovrebbe essere positivo, incoraggiare il bambino che consegue il successo, invece di sottolineare l'insuccesso. Non penalizzate gli errori, perchè questo rende alcuni bambini ansiosi e quindi cade il discorso dell'imparare divertendosi.

Cercate di essere elastici nel vostro appoccio alla scrittura di programmi educativi. Non c'è nessuna ragione particolare, per esempio, per la quale il calcolatore debba sempre generare problemi e controllare le relative risposte. Il nostro prossimo programma, EQUAZIONI, (che genera equazioni del tipo $ax+by=z$) prepara un problema e lo stampa sullo schermo. Quando lo studente vuole vedere la risposta, preme semplicemente un tasto (eccetto BREAK) e la risposta insieme con le variabili sostituite nelle equazioni originali viene stampata. Potreste desiderare di introdurre questo programma nello Spectrum e come esercizio, aggiungere un intero sistema di conversazione, un meccanismo di aggiornamento del punteggio e i modi di accettare e valutare le risposte dello studente.

```
 10 REM EQUAZIONI
 20 INK 7: PAPER 1
 30 BORDER 1: CLS
 40 DEF FN M(N)=INT (RND*N)+1
 50 LET A=FN M(10)
 60 LET B=FN M(10)
 70 LET C=FN M(10)
 80 LET D=FN M(10)
 90 LET X=FN M(10)
100 LET Y=FN M(10)
110 LET E=A*X+B*Y
120 LET F=C*X+D*Y
130 PRINT ''' INK 6;"Le equazio
ni sono:"
140 PRINT 'TAB 8; INVERSE 1;A;"
x + ";B;"y = ";E
150 PRINT 'TAB 8; INVERSE 1;C;"
x + ";D;"y = ";F
160 PRINT '"Le devi risolvere i
n "; INVERSE 1;"x"; INVERSE 0;"
e "; INVERSE 1;"y"
170 PRINT ''"Premi un qualsiasi
 tasto per    vedere la soluzion
e"
```

```
 180 PAUSE 0
 190 CLS
 200 PRINT ''''TAB 8;A;"x + ";B;
"y = ";E
 210 PRINT 'TAB 8;C;"x + ";D;"y
= ";F
 220 PRINT 'TAB 4;"Il valore di
x e' "; FLASH 1;X
 230 PRINT 'TAB 4;"Il valore di
y e' "; FLASH 1;Y
 240 PRINT 'TAB 8;A;"*";X;" + ";
B;"*";Y;" = ";E
 250 PRINT 'TAB 8;C;"*";X;" + ";
D;"*";Y;" = ";F
 260 PRINT '''"Premi un qualsias
i tasto per un altro problema"
 270 PAUSE 0
 280 RUN
```

Il nostro prossimo programma GATTI E COSE é per bambini molto piccoli. Esso usa la funzione INKEY$ dall'inizio alla fine e ha un rigoroso sistema di convalida degli input. Un numero casuale di carri armati, autocarri o gatti è visualizzato sullo schermo e il bambino deve introdurre il numero corretto.

Notate la differenza tra l'uso della INPUT e della INKEY$: INPUT vi dà la possibilità di correggere gli errori, la INKEY$ ha il vantaggio di ottenere degli input di caratteri isolati senza dover premere ENTER ogni volta. Ciò rende il programma più rapido e lo rifinisce in modo più professionale.

I carri armati, gli autocarri e i gatti sono stampati con i simboli grafici A e B definiti dall'utente; i dati necessari sono contenuti nelle linee da 510 a 530. Il tipo di oggetto é selezionato casualmente alla linea 20, la RESTORE calcolata pone il puntatore della DATA alla linea corretta. Le linee da 30 a 70 assegnano il nome dell'oggetto a n$ e predispongono i simboli grafici. Le linee dalla 80 alla 140 scelgono il numero di oggetti, li stampano alla posizione dello schermo specificata dai dati alla linea 600 e chiedono al bambino quanti oggetti vede. Notate che dovete avere il cursore G quando battete AB alla linea 120. BEEP é usato in congiunzione con l'esposizione visiva per dirigere l'attenzione del bambino alla sezione di correzione (linee da 180 a 230). L'elogio, accompagnato da una sequenza crescente di note, viene prodotto dalle linee 250 e 260.

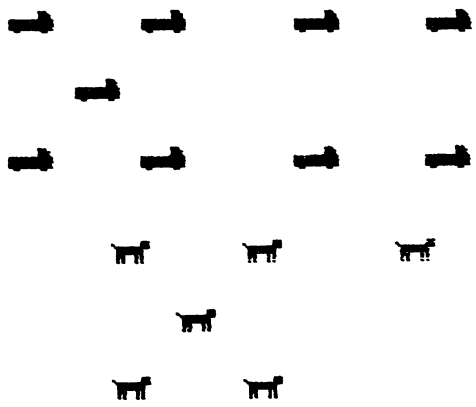
Potreste voler migliorare il programma aggiungendo altre istruzioni per mantenere il punteggio e per stamparlo dopo 10 domande. Potreste anche rendere la visualizzazione un po' più attraente immettendo un colore di INK casuale alla linea 140. Evitate gatti bianchi su schermo bianco in quanto i bambini si sentirebbero imbrogliati non potendoli contare. Provate anche ad aumentare il numero degli oggetti,

cosa ne dite di uccelli e ragni? Date ai bambini la possibilità di inventarne di nuovi. Scrivete linee DATA addizionali a partire dalla 540 ed incrementate il numero alla linea 10, da 3 al totale di tipi di oggetti. Un'altra variante potrebbe essere la disposizione casuale degli oggetti, se non vi piace il modello 'domino'.

All'inizio di questo capitolo ci siamo concentrati su problemi numerici, i più facili da scrivere per lo Spectrum. Ma non c'é nessuna ragione di pensare che quelli che richiedono risposte verbali siano molto più difficili.

```
  5 LET b$="
            "
 10 LET i=1+INT (RND*3)
 15 REM SCEGLI E FORMA GLI OGGE
TTI
 20 RESTORE 500+10*i
 30 READ n$
 40 FOR n=1 TO 2: READ s$
 50 FOR m=0 TO 7
 60 READ a: POKE USR s$+m,a
 70 NEXT m: NEXT n
 75 REM STAMPA IL NUMERO DEGLI
OGGETTI
 80 LET z=1+INT (RND*9)
 90 RESTORE 600
100 FOR n=1 TO z
110 READ x,y
120 PRINT AT x,y;"AB"
130 NEXT n
140 PRINT AT 15,0;"Quanti ";n$;
" sono?"
150 GO SUB 410
160 IF CODE i$<48 OR CODE i$>57
THEN GO TO 150
170 IF i$=STR$ z THEN GO TO 250
175 REM CORREZIONE
180 PRINT AT 15,0;b$: RESTORE 6
00
190 FOR n=1 TO z
200 READ x,y
210 PRINT AT x,y-1; PAPER 6;n
220 BEEP 1,n
230 NEXT n
240 GO TO 280
```

```
 245 REM ELOGIO
 250 PRINT AT 15,0;"Esatto.";b$:
 PRINT AT 7,0; FLASH 1;z
 260 FOR m=0 TO 4 STEP .1: BEEP
0.02,m*m: NEXT m: BEEP 1,20
 270 PRINT AT 7,0;z
 280 PRINT AT 21,0;"Premi ENTER"
 290 GO SUB 410
 300 IF CODE i$<>13 THEN GO TO 2
90
 310 CLS : GO TO 10
 400 REM INPUT DI UN TASTO
 410 IF INKEY$<>"" THEN GO TO 41
0
 420 IF INKEY$="" THEN GO TO 420
 430 LET i$=INKEY$
 440 RETURN
 500 REM DATI
 510 DATA "carri armati","a",7,1
5,7,255,255,127,63,27,"b",192,25
4,192,255,255,255,254,108
 520 DATA "gatti","a",0,128,127,
63,63,56,40,40,"b",28,28,252,240
,240,80,80,80
 530 DATA "autocarri","a",0,0,0,
255,255,255,255,48,"b",28,30,30,
255,255,255,255,12
 600 DATA 4,5,10,5,4,11,10,11,7,
8,4,18,10,18,4,24,10,24
```



Il prossimo programma VOCABOLARIO FRANCESE, memorizza le risposte e le domande nelle frasi DATA e può essere facilmente esteso o adattato a qualsiasi programma tipo quiz. Notate che in questo programma il computer deve avere il CAPS LOCK inserito dato che le domande e le risposte che il calcolatore deve controllare sono tutte in lettere maiuscole. Il programma permette allo studente di scegliere il numero di domande a cui desidera rispondere (linea 80) e questo numero é usato dal ciclo FOR/NEXT alla linea 100.

Forse pensate che in programmi di questo tipo sia importante assicurarsi che la stessa domanda non venga posta più di una volta nella stessa esecuzione. Se é cosi in futuro potrete adattare o usare il meccanismo adottato in questo programma per assicurarsi che il significato della stessa parola venga richiesto per non più di una volta in ogni esecuzione. La linea 30 dimensiona una matrice e come sapete ne azzerra tutti gli elementi. Come potete vedere, nelle frasi DATA ogni parola in italiano é seguita da una parola in francese, che a sua volta é seguita da un numero. Quando viene scelta una coppia di parole (leggendo a caso nelle frasi DATA delle linee da 110 a 130) viene anche assegnato un valore alla variabile X.

Ogni coppia di parole ha un unico numero associato, e questo elemento della matrice viene posto al valore 1 nella linea 160. La linea 150 controlla se questo elemento è a uno prima di stampare la domanda e se è cosi utilizza l'istruzione RESTORE per scegliere un'altra coppia di parole. Sebbene questo significhi impiegare più tempo alla fine del quiz per trovare una coppia di parole non ancora usate, troverete che il risultato aggiunge, più che togliere, qualcosa al programma. Come la maggior parte dei precedenti, il programma replica alle risposte esatte con grande enfasi mentre invece tralascerà ogni commento appariscente alle risposte errate. Infine noterete che questo programma corregge immediatamente lo studente quando sbaglia.

```
  10 REM VOCABOLARIO FRANCESE
  20 LET SCORE=0: PAPER 1: INK 7
: BORDER 1: CLS
  30 DIM A(20)
  40 PRINT '''TAB 6;"Benvenuti in
 Francia"
  50 PRINT TAB 8;"Vocabolario"
  60 PRINT '''TAB 6;"Quante parol
e vuoi tentare?"
  70 PRINT "(da 1 a 20)"
  80 INPUT B
  90 IF B<1 OR B>20 THEN GO TO 8
0
 100 FOR C=1 TO B
 110 FOR D=1 TO INT (RND*20)+1
 120 READ E$: READ F$: READ X
```

```
 130 NEXT D
 140 RESTORE : CLS
 150 IF A(X)=1 THEN GO TO 110
 160 LET A(X)=1
 170 PRINT ''' INK RND*7; PAPER
9;"Domanda numero ";C;": "
 180 PRINT '''"Qual e' la parola
 francese per";TAB 4;E$;"?"
 190 INPUT A$
 200 IF A$=F$ THEN LET SCORE=SCO
RE+1: BEEP .1,2.5*SCORE: PRINT F
LASH 1; INK 2; PAPER 5;'''"Si, "
;F$;" e' giusto"
 210 IF A$<>F$ THEN PRINT ''"No,
 mi dispiace."''"La parola franc
ese per ";E$;TAB 8;"e' ";F$
 220 IF C<8 THEN PRINT '' INK 3;
 FLASH 1; BRIGHT 1;"Il tuo punte
ggio e' ora ";SCORE;" "
 230 PAUSE 150
 240 NEXT C
 250 INK 2: PAPER 7: FLASH 1: CL
S : PRINT AT 6,3;"In questo test
 il tuo punteggio e' stato ";SCO
RE
 260 PAUSE 200: FLASH 0
 270 RUN
 280 DATA "DOVUNQUE","PARTOUT",1
,"RARAMENTE","RAREMENT",2,"SPESS
O","SOUVENT",3,"MAI","JAMAIS",4,
"SEMPRE","TOUJOURS",5
 290 DATA "MOLTO","TRES",6,"PIU'
","PLUS",7,"MENO","MOINS",8,"SI"
,"OUI",9,"NO","NON",10
 300 DATA "GRAZIE","MERCI",11,"U
NO","UN",12,"DUE","DEUX",13,"TRE
","TROIS",14,"QUATTRO","QUATRE",
15
 310 DATA "CINQUE","CINQ",16,"SE
I","SIX",17,"SETTE","SEPT",18,"O
TTO","HUIT",19,"NOVE","NEUF",20
 320 STOP
```

Potrete usare il prossimo programma, SUPER QUIZ, per qualsiasi cosa, dall'esercizio settimanale di ortografia alla biologia universitaria. Il punto è sarete voi a fornire le domande. Questo è più semplice di quanto possiate pensare e inoltre studenti volonterosi faranno la maggior parte del lavoro per voi.

Il programma viene adoperato per memorizzare una serie di domande e/o suggerimenti insieme con le relative risposte.

Qui c'é qualche esempio:

Come si dice "Dare" in Francese?
(Risposta: donner)

Dire il nome del minerale estratto a St. Austell.
(Risposta: Caolino).

Dire il nome della particella subatomica con massa 1 e carica zero.
(Risposta: Neutrone).

Quali cellule contengono i citocromi?
(Risposta: Mitocandri).

Brevi e semplici suggerimenti dovrebbero essere sufficienti per l'esercizio settimanale di ortografia. Se la lista contiene "Cavaliere", un suggerimento adatto potrebbe essere: "uomo armato a cavallo". Convincete i vostri bambini ad introdurre loro stessi le parole e i "suggerimenti" associati e poi provate il test per assicurarvi che sia tutto OK.

Il programma inizia presentando tutte le risposte facendole lampeggiare brevemente, una alla volta, sullo schermo. Quindi presenta le domande una volta sola in ordine casuale. Se una risposta è errata o semplicemente premete ENTER, vi verrà dato un suggerimento, "c.....e" per cavaliere, oppure "d.....r" per donner. RUN 500 vi permette di creare un nuovo set di domande e risposte.

```
  10 REM INIZIALIZZAZIONE
  20 LET s=0: LET x=m: LET b$="-
----------": LET e$="
                          "
  30 FOR n=1 TO m: LET y(n)=n: N
EXT n
  40 GO SUB 400: CLS : FOR n=1 T
O m: PRINT AT 10,8;a$(n): PAUSE
50: CLS : NEXT n
  50 REM PROGRAMMA PRINCIPALE
  60 LET z=INT (RND*x+1): LET q=
y(z)
```

```
  70 LET x=x-1
  80 FOR n=z TO x: LET y(n)=y(n+
1): NEXT n
  90 CLS : LET t=2
 100 PRINT AT 3,0;c$(q): INPUT i
 $
 110 IF i$<>a$(q)( TO l(q)) THEN
 GO TO 240
 120 LET s=s+t: GO TO 210
 130 GO SUB 410: CLS
 140 IF x>0 THEN GO TO 60
 150 REM PUNTEGGIO FINALE
 160 PRINT AT 5,0;"Il tuo punteg
gio e' stato di ";s;" su ";m*2
 170 STOP
 200 REM ELOGIO
 210 PRINT AT 7,0;"Bene, ";i$;"
e' esatto.";AT 9,15;e$
 220 GO TO 130
 230 REM RISPOSTA ERRATA
 240 IF i$="" THEN GO TO 260
 245 PRINT AT 7,0;i$;" e' sbagli
ato.";e$;AT 9,0;e$
 250 GO SUB 410
 260 IF t=2 THEN GO TO 310
 270 PRINT AT 7,0;"La risposta e
satta e':";e$;AT 9,15;a$(q)
 280 GO TO 130
 300 REM SECONDO TENTATIVO
 310 LET t=1
 320 PRINT AT 7,0;"Ti do' un aiu
to: ";e$;AT 21,0;e$
 330 PRINT AT 9,15;a$(q,1)+b$(1
TO l(q)-2)+a$(q,l(q))
 340 GO TO 100
 400 REM ENTER PER CONTINUARE
 410 PRINT AT 21,0;"Premi ENTER
per continuare."
 420 GO SUB 460
 430 IF CODE k$=13 THEN RETURN
 440 GO TO 420
 450 REM INPUT DI UN TASTO
 460 IF INKEY$<>"" THEN GO TO 46
```

```
0
 470 IF INKEY$="" THEN GO TO 470
 480 LET k$=INKEY$: RETURN
 500 REM DOMANDE E RISPOSTE
 510 PRINT "Quante domande?": IN
PUT m
 520 DIM c$(m,64): DIM a$(m,15):
 DIM l(m): DIM y(m)
 530 FOR q=1 TO m
 540 CLS : PRINT "Domanda ";q''
 550 PRINT "Introduci la domanda
o il suggerimento.": INPUT c$(q)
 560 PRINT 'c$(q)''"Introduci la
 risposta.": INPUT i$
 570 PRINT 'i$;AT 20,0;"Se va  b
ene premi s;"," per cancellare p
remi n."
 580 GO SUB 460
 590 IF k$="n" THEN GO TO 540
 600 IF k$<>"s" THEN GO TO 580
 610 LET l(q)=LEN i$: LET a$(q)=
i$
 620 NEXT q
 650 REM MEMORIZZAZIONE
 660 CLS : INPUT "Quale nome vuo
i usare?";p$
 670 SAVE p$ LINE 20
```

Per capire come funziona questo programma è meglio analizzarlo a partire dalla linea 500, dove vengono inserite le domande e le rispostre. Il numero totale delle domande, m, ottenuto alla linea 510 è usato per dimensionare quattro matrici alla linea 520. La matrice c$ contiene m domande, ciascuna delle quali di massimo 64 caratteri. La matrice a$ contiene le m risposte di massimo 15 caratteri. Se volete domande o risposte più lunghe non avete che da cambiare queste frasi DIM. La matrice 1 contiene la lungezza reale di ogni risposta, dato che le stringhe delle risposte sono completate con spazi quando vengono memorizzate nella matrice. La lunghezza delle risposte viene usata quando la risposta del bambino è confrontata con quella corretta.

Le risposte vengono inizialmente accettate usando INPUT i$, in modo che le loro lunghezze possano essere determinate prima di venir memorizzate nella matrice a$.

La linea 660 e la 670 vi obbligano a salvare il programma immediatamente. Una

volta che avete memorizzato tutte le vostre domande e le vostre risposte nelle matrici, verranno cancellate dalla memoria se usate RUN.

Se volete far girare il programma un'altra volta, dovete usare GO TO 20, manterrete così intatte le domande e le risposte. Vi potrebbe interessare scrivere alcune linee alla fine del programma principale per darvi la possibilità di farlo girare ancora.

Ritornando all'inizio del programma, le linee 20 e 30 inizializzando le variabili; s è il punteggio e x è il numero di domande che devono essere ancora poste; b$ é usato per i trattini nei suggerimenti ed e$ é usato per cancellare singole linee sullo schermo.

La linea 40 fa lampeggiare tutte le risposte sullo schermo. Questo è un valido strumento per esercitare la memoria con piccoli gruppi di domande, ma dovreste eliminarlo quando il numero di domande è troppo elevato.

La matrice y (dimensionata alla linea 520) memorizza i numeri delle domande in modo da evitare noiose ripetizioni in fase di esecuzione. La linea 30 inizializza la matrice in modo tale che y(1)=1, y(2)=2 e così via. Se vi fossero solo quattro domande nel test, la linea 60 darebbe a z un valore compreso tra 1 e 4. Supponiamo che z sia uguale a 2, il numero di domanda q viene posto uguale a y(2), che è 2. La linea 70 decrementa x, il numero delle domande non ancora poste. La linea 80 elimina i numeri delle domande che sono state usate:

Prima della linea 80	Dopo la linea 80
y(1) = 1	y(1) = 1
y(2) = 2	y(2) = 3
y(3) = 3	y(3) = 4
y(4) = 4	y(4) = 4

Otterrete due punti se risponderete correttamente al primo tentativo e un punto se lo farte al secondo tentativo. La linea 110 usa la lunghezza della risposta 1(q) per rimuovere gli spazi in più, prima di memorizzarla nella matrice a$. Dovete rimuovere questi spazi perchè:

"Costantinopoli" <> "Costantinopoli "

La lunghezza della risposta é usata ancora alla linea 330 per ottenere da b$ il numero di trattini da usare nel suggerimento "c.....".

Potreste usare questo programma per far ripassare ai vostri figli il lavoro scolastico di diverse materie. Se introdurrete nello Spectrum un vocabolario francese, definizioni di chimica, e domande sulla letteratura, otterrete presto un archivio di varie materie (e fra l'altro ripasserete anche voi senza accorgervene). Lo scorrere preliminare delle risposte e i suggerimenti rendono facile l'apprendimento, se il programma viene usato a intervalli regolari durante l'anno. Questo darà al bambino confidenza con il significato fondamentale delle parole, senza il quale non vi possono essere

miglioramenti in nessuna materia. Per migliorare il programma si potrebbe aggiungere un'altra matrice per memorizzare spiegazioni o informazioni che potrebbero essere richiamate dall'utente. Un altro possibile sviluppo potrebbe essere quello di controllare le risposte errate con una lista di termini e far stampare allo Spectrum il corretto significato della risposta del bambino.

Da questo programma educativo di tipo generale passiamo ad un altro più specifico. Questo che mi é venuto in mente leggendo il libro "The Standard Reading Tests", (consultate i suggerimenti per ulteriori letture alla fine di questo capitolo) mette alla prova la velocità di lettura e di ricezione.

Il calcolatore sceglie una frase tra quelle memorizzate con le DATA a partire dalla linea 160 e la stampa sullo schermo. Questa frase verrà mantenuta in visione sullo schermo per un tempo relativo alla sua lunghezza (vedere linea 60) e quindi sarà cancellata; ora viene chiesto allo studente di riscrivere quanto che ha appena letto sullo schermo. La risposta dello studente viene confrontata (fatta eccezione per la prima lettera che può essere sia minuscola che maiuscola) con la frase generata dal calcolatore.

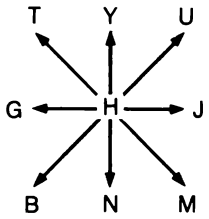
Se le due frasi sono uguali viene stampato il messaggio "Ben fatto, é giusto", poi lo schermo viene ripulito e viene proposta un'altra frase. Altrimenti la frase da indovinare viene ripetuta dopo una breve pausa e il ciclo si ripete finchè la frase non è introdotta correttamente.

Dovreste alterare il 20 alla fine della linea 60 per adeguare la velocità del programma alla capacità di lettura del bambino. Inoltre anche le frasi dovrebbero adattarsi all'abilità e all'età del bambino. Se lo desiderate potete anche sostituirvi al calcolatore e farvi ripetere la frase dal bambino, se questo vi sembra un modo migliore di controllare la sua velocità di lettura. Comunque il libro che ho menzionato contiene una ricca varietà di frasi e parole che si possono usare in programmi del genere fatti da voi.

```
  10 INK 1: PAPER 7: BORDER 7: C
LS
  20 PAUSE 50: RESTORE
  30 FOR J=1 TO INT (RND*11)+1
  40 BEEP 1/J,3*J: READ A$: NEXT
 J
  50 PRINT AT 10,0;A$
  60 PAUSE (LEN A$)*20
  70 CLS
  80 INPUT INK 2;"Ora introduci
la frase che hai  appena letto",
,B$
  90 PRINT INVERSE 1;" ";B$;" "
 100 PAUSE 50
```

```
 110 IF A$(2 TO )=B$(2 TO ) THEN
 BEEP .3,LEN A$: PRINT '' INK 3;
 FLASH 1;"Ben fatto, e' giusto":
 PAUSE 300: RUN
 120 PRINT ''"Spiacente e' errat
o"
130 PRINT ''"Da capo"
 140 PAUSE 200: CLS
 150 GO TO 50
 160 DATA "Guarda il cane che do
rme"
 170 DATA "Il gatto ha quattro z
ampe"
 180 DATA "L'uovo e' cotto"
 190 DATA "Il sole scotta"
 200 DATA "La volpe corre veloce
"
 210 DATA "Il papero nuota lenta
mente"
 220 DATA "Il cammello ha una go
bba"
 230 DATA "Il campanello sta suo
nando"
 240 DATA "Il pesce nuota"
 250 DATA "Il libro e' aperto"
 260 DATA "La gallina canta"
```

I bambini gradiranno molto l'uso del prossimo programma per creare schizzi e ghirigori, ma noi lo useremo per disegnare mappe e diagrammi. Potete disegnare segmenti orizzontali, verticali e diagonali usando questi tasti:



Il tasto di 'base' H (home) al centro non ha effetto.

```
  10 BORDER 6
  20 INPUT "Coordinata X inizial
e?";x
  30 INPUT "Coordinata Y inizial
e?";y
  40 PLOT x,y
  70 LET c$=INKEY$
  80 LET x=x+(c$="u")+(c$="j")+(
c$="m")-(c$="t")-(c$="g")-(c$="b
")
  90 LET y=y+(c$="t")+(c$="y")+(
c$="u")-(c$="b")-(c$="n")-(c$="m
")
 100 LET x=x-(x>255)+(x<0)
 110 LET y=y-(y>175)+(y<0)
 120 IF c$="s" THEN STOP
 130 GO TO 40
```

Scrivete il programma e fatelo eseguire. Selezionate la posizione di partenza tramite le coordinate x e y, come mostrato nel Manuale dello Spectrum (pagina 102 della prima edizione). La linea 70 controlla quale tasto state premendo e quindi lo memorizza in c$. Le linee 80 e 90 cambiano le coordinate x e y in conseguenza del tasto premuto. Se premete 'm' (c$="m") é verificata e assume il valore 1.

Le rimanenti espressioni logiche delle linee 80 e 90 sono false, quindi assumono il valore 0.

Vengono svolte le operazioni x=x+1 e y=y+1 e il prossimo sarà stampato verso sud-ovest rispetto al precedente. Le linee 100 e 110 prevengono l'eventualità che usciate dalla superficie dello schermo; questo darebbe luogo ad un errore. La linea 120 vi permette di fermarvi premendo 's'.

Potrete salvare i vostri lavori in questo modo: SAVE "schizzo" SCREEN$. Questo semplice programma può essere esteso in vari modi. Potreste ad esempio riscrivere la linea 120, così:

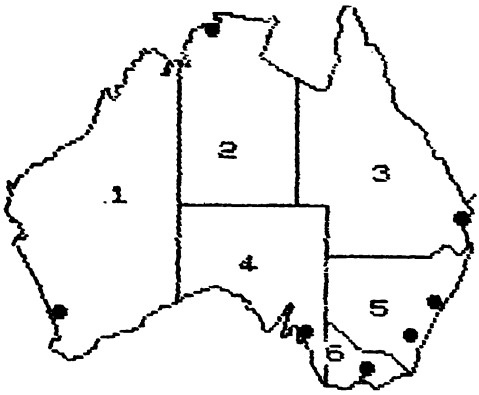
```
120 IF c$="c" THEN GO TO 140
```

"c" sta per cambio, passa cioè il controllo alla linea 140.
Qui potete introdurre frasi del tipo:

```
INPUT "colore di INK"; ink
```

Scrivete INK ink alla linea 40 dopo il comando PLOT, ciò vi permetterà di disegna-

re a colori. Tutto questo potrebbe essere fatto anche per comandi del tipo: PAPER, OVER, FLASH, BRIGHT e INVERSE. I risultati ottenibili possono essere molto elaborati.



Un interessante modifica al programma potrebbe essere:

```
50 IF INKEY$<>"" THEN GO TO 50
60 IF INKEY$="" THEN GO TO 60
```

Queste due linee rallentano il programma in modo che il punto venga disegnato soltanto quando premete un tasto.

Se volete disegnare una mappa dovrete trovarne una che si adatti alle dimensioni del vostro schermo. Tracciate la mappa su un foglio di carta trasparente e adagiatelo sullo schermo del televisore. Usate il programma per copiare la mappa, lasciando un po' di spazio per le domande e le risposte.

Quando il vostro disegno é completato, salvatelo. Potete aggiungere qualcos'altro al disegno, ad esempio un cerchietto che rappresenti una città: CIRCLE 55,50,2. Per trovare le coordinate dei punti sullo schermo procedete in questo modo: fate eseguire il programma un'altra volta, tracciate una linea fino al punto richiesto, fermate il programma e fatevi stampare le coordinate x e y in un angolo dello schermo.

Quindi fate ripartire il programma con GO TO 40 e ripetete il procedimento. Infine ricaricate una copia pulita della mappa, aggiungete le città e salvate la versione definitiva.

Potete applicare la stessa tecnica per produrre pressapoco qualsiasi cosa vogliate, dai diagrammi di un circuito in fisica a sezioni trasversali di tronchi d'albero in biologia.

Il prossimo lavoro è collegare la vostra mappa o il vostro diagramma con un programma di domande e risposte. Se usate ad esempio il programma SUPER QUIZ, esso avrà bisogno di opportune modifiche. Rimpiazzate la linea 40 con:

```
40 LOAD "mappa" SCREEN$
```

Dovreste inoltre scrivere tutte le frasi PRINT in modo che agiscano sul fondo dello schermo.

La seguente subroutine dovrebbe aiutarvi a farlo:

```
350 REM PULITURA DELLE ULTIME TRE LINEE
360 PRINT AT 19,0;e$;e$;e$
370 PRINT AT 19,0;
380 RETURN
```

Ora date il comando RUN 500 e introducete le vostre domande e risposte. Quando lo avrete salvato su nastro, premete BREAK e ricaricate la vostra mappa. Salvatela usando SAVE "mappa" SCREEN$ sullo stesso nastro immediatamente dopo il programma di domande e risposte. Ora ricaricate il programma SUPER QUIZ, che caricherà automaticamente la mappa e andrà avanti con le domande.

Come avete visto in questo capitolo abbiamo sviluppato una serie di programmi che molto probabilmente gli studenti apprezzeranno e che richiedono poco tempo per essere scritti. Questi programmi appartengono, per la maggior parte, alla categoria degli esercizi pratici e di apprendimento.

Uno studente, per imparare efficacemente, ha bisogno di: molta pratica; sapere

immediatamente se le sue risposte sono giuste o sbagliate; la possibilità di stabilire il proprio ritmo di studio e cosa più importante la soddisfazione che deriva dalla riuscita.

Ci sono altri possibili programmi educativi, come quelli che richiedono una scelta tra più risposte proposte e quelli basati sulle figure; ad esempio quelli che trattano di geometria, di cui non abbiamo parlato. Comunque il materiale che vi abbiamo presentato dovrebbe darvi un buon punto di partenza per sviluppare il vostro software.

Se vi sentite ancora a corto di idee, potreste dare un'occhiata ai cataloghi di programmi educativi di qualsiasi computer. La semplice lettura del catalogo dovrebbe essere sufficiente per farvi scaturire più idee di quante ne possiate realizzare (infatti questo è uno dei modi che uso per farmi venire delle idee per i programmi in generale, leggendo cioé le descrizioni di ciò che fanno i programmi di altri quando vengono eseguiti).

Adesso che siamo arrivati alla fine di questo capitolo, dovreste possedere un numero di semplici ma efficaci programmi educativi. Quando estenderete la vostra collezione, sia con software prodotto da voi, che acquistato, controllate che siano soddisfatte le seguenti condizioni:

- Non dovrete essere obbligati a procedere attraverso lunghe descrizioni ogni volta che fate girare il programma.
- L'ammontare del materiale sullo schermo dovrà essere sempre reso minimo in modo tale che l'attenzione sia focalizzata sui dettagli importanti.
- Nella presentazione del testo dovrebbero essere usati con criterio colori differenti, come nel programma delle divisioni.
- Non dovrebbe essere possibile bloccare il programma, comunque rispondiate alle domande.
- Se la risposta è sbagliata, la risposta esatta deve essere chiaramente evidenziata.
- Dovreste essere in grado di lavorare con il programma alla vostra velocità senza che le informazioni spariscano prima che possiate leggerle e senza periodi di forzata inattività.

In sostanza un buon programma è quello che raggiunge i suoi obiettivi in modo piacevole.

SUGGERIMENTI PER ULTERIORI LETTURE:

- *"The Computer Tutor"* – Orwig Gary W. e Hodges William S. (Winthrop Publishers Ink, USA 1981)

— *"The Standard Reading Tests"* — Daniels J.C. e Diack Hunter (Chatto Educational Ltd., 1958)

— *"Problems For Computer Solution"* — Rogowski Stephen J. (Creative Computing Press, Usa 1979)

— *"Microcomputers In The Classroom"* — Maddison Alan (Holder and Stoughton, London 1982)

— *"Understanding Calculator Math"* — Oliva Ralph A. (Texas Instruments Inc, Dallas-Texas; distribuito da Tandy/Radio Shack, 1978).

CAPITOLO 6

GIOCARE CON LO SPECTRUM

Qualunque sia il motivo per il quale avete comprato lo Spectrum, è probabile che impiegherete parte del vostro tempo giocando con lui. È anche probabile che presto vi accorgerete che una delle più grandi soddisfazioni quando si possiede un calatore; è inventare i vostri programmi, giochi o altro, e farli eseguire.

In questo capitolo daremo un'occhiata a un certo numero di giochi. Tutti verranno esaminati in dettaglio con l'intenzione di darvi qualche suggerimento sulla scrittura dei giochi, che potrebbe risultarvi utile quando deciderete di crearli per vostro conto.

Posso immaginare che siate tentati di introdurre subito la versione finale di ogni gioco, senza leggere prima il materiale che precede e segue il listato.

Se fate così avrete certamente un programma che funziona, ma non raggiungerete lo scopo principale di questo capitolo.

Provate a trattenervi, e seguite le descrizioni linea per linea; per quei programmi che sono introdotti gradualmente scrivete ogni parte solo quando avete raggiunto la descrizione.

I primi due programmi, BOMBARDAMENTO e JACK-MAN, sono spiegati molto più in dettaglio rispetto agli altri. Sono quelli che dovreste leggere più attentamente, anche se decidete di saltare le spiegazioni degli altri. Ci sono molte idee nei primi due programmi che ritengo siano utili da conoscere, non solo per comprendere quei particolari programmi, ma anche per applicarle ad altri problemi di programmazione.

BOMBARDAMENTO

Il primo gioco riguarda un aeroplano che sorvola una città cercando di ridurre i grattacieli a un cumulo di macerie. L'aereoplano si muove attraverso lo schermo abbassandosi sempre di più.

Alla fine si schianterà contro un grattacielo se non avete distrutto gli edifici.

Il nostro primo problema è stampare i grattacieli. Questo si può fare abbastanza facilmente con un ciclo FOR/NEXT da 0 a 31 (le colonne dello schermo), con un ciclo nidificato al suo interno che stampa i grattacieli a blocchi. Per capire questo concetto provate questo programma:

```
10 FOR a=0 TO 31
20 FOR b=11 TO 21
30 PRINT AT b,a;"■"
40 NEXT b
50 NEXT a
```

Esso stampa un gran numero di grattacieli di uguale lunghezza sullo schermo. Provate ad alterare i valori di b alla linea 20 per variare l'altezza dei grattacieli.

Comunque questa routine non produce un orizzonte molto interessante; il risultato assomiglia di più a una grossa scatola di scarpe che a una città. Per dare ai grattacieli altezze variabili abbiamo bisogno di un fattore casuale. Provate a cambiare la linea 20 con:

```
20 FOR b = INT (RND*22) TO 21
```

L'orizzonte è ora molto più vario. La routine comincia a creare qualcosa che assomiglia a una città. Comunque manca ancora di uniformità. La prima routine crea una città che è troppo uniforme, la seconda ne crea una troppo casuale. Abbiamo bisogno di una traccia su cui può lavorare il fattore casuale. Dobbiamo essere noi a controllare il fattore casuale e non viceversa. A questo scopo possiamo introdurre un "fattore di difficoltà".

Per disegnare una città che abbia un aspetto convincente abbiamo bisogno di un'altezza approssimativa attorno alla quale costruire grattacieli. Alcuni saranno più alti dell'altezza media, altri più bassi, ma nessuno sarà molto più alto o molto più basso. Nell'ultima routine l'altezza media era di 11 caratteri; ma alcuni edifici erano alti 21 caratteri, mentre altri solo uno. Un gioco difficile ha un'altezza media intorno ai 18 caratteri, così l'aereoplano si disintegra molto presto. Un gioco più facile dovrebbe avere un'altezza media attorno ai cinque caratteri. Il vostro aereoplano disporrebbe di un tempo abbastanza lungo prima di raggiungere quel livello di un tempo abbastanza lungo prima di raggiungere quel livello e sfasciarsi contro gli edifici. Ovviamente in un gioco facile sarebbe assurdo avere un'altezza media di cinque e uno o due edifici alti 20 caratteri, perché il gioco non durerebbe più abbastanza a lungo per essere considerato facile.

All'inizio del gioco chiederemo all'utente un grado di difficoltà (ad esempio nella

```
10 INPUT "Difficolta' (1-9)",d
20 IF d<1 OR d>9 THEN GO TO 10
30 LET d=12-d
35 CLS
40 FOR a=0 TO 31
50 FOR b=d+INT (RND*d) TO 21
60 PRINT AT b,a;"■"
70 NEXT b
80 NEXT a
90 GO TO 10
```

fascia da uno a nove). Con questo potremmo costruire grattacieli di altezza approssimativa basata sul livello di difficoltà scelto. Provate la seguente routine:

Introducete un numero da uno a nove e lo Spectrum stamperà un orizzonte abbastanza realistico. Provate con un numero basso e noterete che la città risulta anch'essa più bassa. Cosa accade quando introducete un numero che è fuori dalla fascia richiesta? Perché? Ogni programma che scrivete dovrebbe contenere una routine che rifiuta gli input inaccettabili.

Molto spesso le persone commettono errori quando introducono delle informazioni attraverso la tastiera; se non controllati questi errori possono bloccare il programma oppure, se l'esecuzione prosegue, possono confondervi con i risultati che generano. Una sola linea che controlla la dimensione del numero introdotto può risolvere il problema.

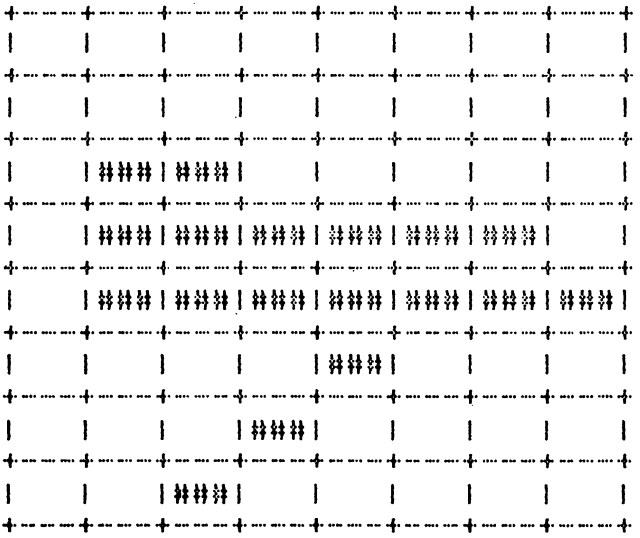
Se avete capito cosa fa il programma fino a questo punto, siamo pronti per proseguire con il passo successivo: produrre un aereoplano in movimento che sorvola la città.

Prima di tutto dobbiamo scegliere la forma dell'aereoplano. Osservando la tastiera non ho trovato nessun simbolo che assomigli, anche solo vagamente, a un velivolo di qualunque tipo. Sembra proprio che dovremo usare la grafica definibile dall'utente. Ecco spiegato passo per passo come procedere:

1) Disegnare una scacchiera otto per otto

2) Riempire i riquadri fino a creare qualcosa che assomiglia a un aeroplano visto di profilo. Non preoccupatevi se vi sembra un po' squadrato, probabilmente visto sullo schermo migliorerà di molto.

Questo è quello che ho usato io:



Adesso riscrivete tutte le righe usando uno zero se il riquadro è vuoto e un uno se è pieno. Per il carattere precedente avreste ottenuto:

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 1 1 0 0 0 0 0

0 1 1 1 1 1 1 0

0 1 1 1 1 1 1 1

0 0 0 0 1 0 0 0

0 0 0 1 0 0 0 0

0 0 1 0 0 0 0 0

Ogni riga di zero e uno è chiamata numero binario.

3) Usando il vostro Spectrum scrivete per ogni riga:

PRINT BIN (la serie di zero e uno)

e annotatevi il numero decimale che viene stampato a fianco di ogni numero binario. Ad esempio per la terza riga dovreste scrivere:

PRINT BIN 01100000

e il computer dovrebbe stampare il numero 96. Annotatevi questo numero su un pezzo di carta in questo modo:

0 1 1 0 0 0 0 0 96

Non dimenticate gli zeri in fondo, perché:

0 1 1 0 0 0 0 0 = 96

ma

0 1 1 0 0 0 = 24

Ora dovreste avere otto numeri. Se avete usato l'esempio precedente avrete i numeri 0,0,96,126,127,8,16,32 in questo preciso ordine.

Azzerate la memoria del vostro Spectrum (usando il comando NEW) e introducete:

```
10 DATA (i vostri otto numeri)
20 FOR a = 0 TO 7
30 READ b: POKE USR "a"+a,b
40 NEXT a
```

La frase DATA dovrebbe contenere i vostri otto numeri separati da virgole. Fate girare questo programma e nulla sembra accadere. Comunque entrate in modo grafico e introducete la lettera 'a' ed ecco il vostro piccolo aereo. Adesso potete cancellare il programma con NEW se volete e l'aereoplano resterà nel computer pronto per l'uso; l'unico modo per liberarsi di lui è togliere la corrente oppure definire un nuovo carattere sullo stesso tasto. Potete ottenere l'aereoplano usando 'CHR$ 144' oltre che con la 'a' grafica. Questo metodo può essere usato per qualunque carattere vogliate disegnare.

Ora che disponiamo del nostro aereoplano possiamo procedere con la parte del movimento. Introducete l'ultima routine che costruisce la città e aggiungete le seguenti linee (alla linea 20 CHR$ 144 rappresenta il vostro aereoplano)

```
 90 LET u=0
100 LET p=0
140 PRINT AT u,p;CHR$ 144
160 LET p=p+1
170 PRINT AT u,p-1;" "
180 IF p=32 THEN LET p=0: LET u=u+1: BEEP .1,u
190 GO TO 140
```

Vi verrà chiesto il grado di difficoltà e appena lo avrete introdotto vedrete stampata la città. Poi vedrete un piccolo aereoplano 'volare' attraverso lo schermo e abbassarsi sempre di più, nello stesso tempo potrete udire un segnale sonoro (le note sono sempre crescenti per aumentare la tensione). Ben presto l'aereoplano passerà attraverso un edificio e proseguirà fino a uscire dallo schermo; a questo punto il programma si ferma con un messaggio di errore.

Ciò che vogliamo ora è una linea che controlli dove viene stampato l'aereoplano e se ci si accorge che è finito contro un grattacielo lo faccia esplodere. Per questo abbiamo bisogno di due routine: una che controlla l'eventuale disastro e l'altra che stampa l'esplosione.

Consideriamo prima la routine che controlla il verificarsi dell'incidente, ci sono due modi per costruirla:

SCREEN$

Rimpiazzate la linea 60 con:

60 PRINT AT b,a; INVERSE 1;"X" (X Maiuscolo)

e la linea 140 con:

140 IF SCREEN$ (u,p)="X" THEN STOP

SCREEN$ può leggere i caratteri e quindi il programma si ferma se l'aereoplano colpisce una 'X'. Questo significa che la routine di stampa dei grattacieli deve essere modificata per far stampare le 'X' inverse al posto dei quadratini grafici, che non possono essere letti dall'istruzione SCREEN$.

ATTR:

Per usare la funzione ATTR rimpiazzate la linea 60 con:

```
60 PRINT AT b,a; INK 5; PAPER 0;"quadratino grafico"
```

e sostituite la linea 140 con:

```
140 IF ATTR(u,p)=5 THEN STOP
```

In questo modo il programma viene fermato ogni volta che l'aereoplano attraversa qualunque carattere verde su sfondo nero, che non stia lampeggiando e che non sia luminoso. Lo svantaggio di questo metodo è che è un po' complicato e, se in altri programmi ci sono molti caratteri diversi riuniti in una piccola area, si crea un po' di confusione nel tentativo di controllare le caratteristiche di ogni carattere.

Se volete fermare il programma quando l'aereoplano colpisce un carattere giallo lampeggiante dovete cambiae la linea 140 con:

```
140 IF ATT(u,p)=134 THEN STOP
```

Riuscite a capire come funziona?

128 se il carattere sta lampeggiando, altrimenti 0:	128
64 se il carattere è luminoso, altrimenti 0:	0
8 X il colore dello sfondo (nero=0)	0
1 X il colore del testo (giallo=6)	6
	TOTALE = 134

Fra i due metodi, il più semplice è quello che fa uso dell'istruzione SCREEN$:

```
PRINT AT b,a; INVERSE 1;"X"
IF SCREEN$(u,p)="X" THEN . . . .
```

Poichè ci sono solo due cose attraverso le quali può trovarsi il nostro aereoplano, l'aria e i grattacieli, questo metodo soddisfa pienamente le nostre necessità. Rimpiazzate la linea 140 con:

```
140 IF SCREEN$(u,p)="X" THEN STOP
```

aggiungete la linea 142:

```
142 PRINT AT u,p; CHR$ 144
```

e non dimenticate la linea 60:

```
60 PRINT AT b,a; INVERSE 1;"X"
```

Ora il programma si fermerà quando l'aereoplano colpisce un edificio.

Uno STOP è piuttosto brutto e manca privo di immaginazione. Ciò che ci occorre è un'esplosione eccitante per quando l'aereoplano si schianta. Provate questo:

```
200 LET x=p*8: LET y=(21-u)*8
205 FOR a=1 TO 40
207 PLOT x,y
210 DRAW INT (RND*256)-x,INT (RND*158)-y
220 BEEP .1,20: BEEP .01,10
230 NEXT a
```

Scegliete un punto sullo schermo e assegnate le coordinate ad u e p, ad esempio LET u=11 : LET p=16. Introducete queste istruzioni in modo diretto. Adesso scrivete GO TO 200 e premete ENTER.

Dovreste vedere il punto sullo schermo 'esplodere', accompagnato da un rumore appropriato. Tutto quello che dobbiamo fare ora è cambiare la linea 140 con:

```
140 IF SCREEN$(u,p)="X" THEN GO TO 200
```

Fate girare l'intero programma e otterrete:

a) una città stampata sullo schermo
b) un aereoplano che si muove abbassandosi sempre di più e
c) l'aereoplano che colpisce un edificio ed esplode.

Ora leggete ancora il programma. Assicuratevi di aver capito come funziona. Adesso siamo pronti per aggiungere le parti finali. Queste comprendono una bomba che potete lanciare dall'aereo per distruggere i grattacieli per prolungare il vostro volo; inoltre dovremo rendere il programma più professionale.

Sfortunatamente non è mai verosimile pensare di aver completato un gioco. Non appena lo avrete memorizzato e vi sarete seduti ad ammirare quella che ritenete la

versione definitiva, giungerà qualcuno che si metterà a giocare e vi darà una grande idea su come migliorare il vostro lavoro.

Comunque, prima di preoccuparci di questo, dobbiamo aggiungere la parte della bomba al programma. Per questo abbiamo bisogno di un 'flag'. Un flag è un indicatore dello stato di qualcosa. Di solito gli viene assegnato il valore 1 per indicare un si e il valore 0 per indicare no. Nel nostro caso il flag, che chiameremo 'f', sarà uguale a 1 quando la bomba è in aria e 0 quando non è così. All'inizio dobbiamo assegnare il valore 0 perchè la bomba non sta cadendo. Ecco come appare il programma fino a questo momento con f posto a zero:

```
  10 INPUT "Difficolta' (1-9)",d
  20 IF d<1 OR d>9 THEN GO TO 10
  30 LET d=12-d
  35 CLS
  40 FOR a=0 TO 31
  50 FOR b=d+INT (RND*d) TO 21
  60 PRINT AT b,a; INVERSE 1;"X"
  70 NEXT b
  80 NEXT a
  90 LET u=0
 100 LET p=0
 110 LET f=0
 140 IF SCREEN$ (u,p)="X" THEN G
O TO 200
 142 PRINT AT u,p;CHR$ 144
 160 LET p=p+1
 170 PRINT AT u,p-1;" "
 180 IF p=32 THEN LET p=0: LET u
=u+1: BEEP .1,u
 190 GO TO 140
 200 LET x=p*8: LET y=(21-u)*8
 205 FOR a=1 TO 40
 207 PLOT x,y
 210 DRAW INT (RND*256)-x,INT (R
ND*158)-y
 220 BEEP .1,20: BEEP .01,10
 230 NEXT a
```

La bomba è 'disattivata' (f=0). Ciò di cui abbiamo bisogno è un mezzo per consentire all'utente di lanciare la bomba (rendere f=1). Un modo facile potrebbe essere quello di aggiungere questa linea:

```
148 IF INKEY$<>"" THEN LET f=1
```

INKEY$ contiene il carattere del tasto che viene premuto (esso è vuoto se nessun tasto è stato premuto). La routine cambia il valore di f in 1 quando premete un tasto qualunque, cioè quando INKEY$ non è uguale a " ". Potremmo cambiare un po' la routine, in modo che la bomba venga lanciata solo quando viene premuto il tasto zero. Per fare questo abbiamo bisogno di:

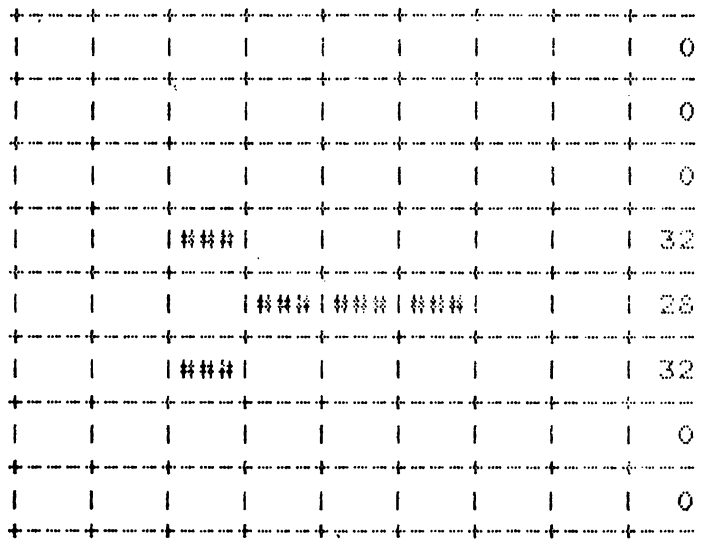
```
148 IF INKEY$="0" THEN LET f=1
```

Comunque è più facile giocare (e ricordarsi come si fa) se potete lanciare la bomba premendo un tasto qualunque, così lasceremo la linea 148 in modo da permettervi di usare qualunque tasto.

Ora ci servono due variabili per memorizzare la posizione della bomba sullo schermo. Quando la bomba è disattivata, essa si trova nell'aereo pronta per essere lanciata:

```
144 IF f=0 THEN LET a=u: LET t=p ('a' e 't' sono le coordinate della bomba)
```

Il prosimo problema è trovare un carattere che assomiglia a una bomba. Un puntino potrebbe servire allo scopo, ma forse preferite definirvi la vostra bomba. Usando il metodo descritto precedentemente, provate a disegnare il carattere della bomba. La sagoma di una bomba potrebbe essere:



La linea che definisce il carattere appare come:

```
POKE USR "b"+a,b
```

e non come:

```
POKE USR "a"+a,b
```

perchè questo sovrappone la bomba all'aereoplano.

Per usare questa bomba premete la b grafica oppure usate CHR$ 145. Se preferite la vita comoda potete usare un puntino come bomba. Dovrete allora scrivere un puntino ogni volta che ho usato CHR$ 145.

Preparata la sagoma della nostra bomba, siamo pronti per stamparla realmente. Siccome la bomba si trova esattamente nella posizione dell'aereoplano, quando non sta cadendo (f=0), non dobbiamo stamparla sino a quando non inizia a cadere, o la bomba cancellerà l'aereoplano. La linea 158 stampa la bomba:

```
158 IF f=1 THEN PRINT AT a, t; CHR$ 145: BEEP .01,60-a
```

Ora la bomba verrà stampata ogni volta che premete un tasto. Tuttavia non cadrà fino a quando non aggiungerete la linea 155 che stampa un blank sulla vecchia posizione della bomba e incrementa la variabile a per realizzare il movimento:

```
155 IF f=1 THEN PRINT AT a, t; " ": LET a=a+1
```

Sebbene ora disponiamo di una bomba in movimento, essa non è molto valida, perchè precepita sul fondo dello schermo e arresta il programma con un messaggio di errore. Ci serve quindi una linea che controlli se la bomba ha colpito l'edificio (sul tipo della linea 140), o se ha raggiunto la fine dello schermo.

```
157 IF SCREEN$ (a, t) ="X" OR a=21 THEN GO TO 300
```

Ora possiamo scrivere a partire dalla linea 300 una routine che faccia esplodere l'edificio:

```
300 FOR a=a TO 21
310 IF RND>.99 THEN GO TO 340
315 BEEP .005,a-20
320 PRINT AT a,t;" "
```

```
330 NEXT a

340 LET f=0: GO TO 140
```

Tutto quello che ci serve adesso è:

```
235 GO TO 10
```

e avremo un gioco completo. È abbastanza difficile mirare e riuscire a lanciare la bomba sugli edifici che volete distruggere.

Ecco un listato completo del gioco:

```
  10 INPUT "Difficolta' (1-9)";d
  20 IF d<1 OR d>9 THEN GO TO 10
  30 LET d=12-d
  35 CLS
  40 FOR a=0 TO 31
  50 FOR b=d+INT (RND*d) TO 21
  60 PRINT AT b,a; INVERSE 1;"X"
  70 NEXT b
  80 NEXT a
  90 LET u=0
 100 LET p=0
 110 LET f=0
 140 IF SCREEN$ (u,p)="X" THEN G
O TO 200
 142 PRINT AT u,p;CHR$ 144
 144 IF f=0 THEN LET a=u: LET t=
p
 148 IF INKEY$<>"" THEN LET f=1
 155 IF f=1 THEN PRINT AT a,t;"
": LET a=a+1
 157 IF SCREEN$ (a,t)="X" OR a=2
1 THEN GO TO 300
 158 IF f=1 THEN PRINT AT a,t;CH
R$ 145: BEEP .01,60-a
 160 LET p=p+1
 170 PRINT AT u,p-1;" "
 180 IF p=32 THEN LET p=0: LET u
=u+1: BEEP .1,u
 190 GO TO 140
```

```
 200 LET x=p*8: LET y=(21-u)*8
 205 FOR a=1 TO 40
 207 PLOT x,y
 210 DRAW INT (RND*256)-x,INT (R
ND*158)-y
 220 BEEP .1,20: BEEP .01,10
 230 NEXT a
 235 GO TO 10
 300 FOR a=a TO 21
 310 IF RND>.99 THEN GO TO 340
 315 BEEP .005,a-20
 320 PRINT AT a,t;" "
 330 NEXT a
 340 LET f=0: GO TO 140
```

Il gioco funziona, ma mancano diverse cose. Nella sua forma attuale non c'è il colore e non viene tenuto il punteggio. Provate ad aggiungere queste parti al programma (potete usare una variabile s ed incrementarla ogni volta che un carattere di un grattacielo è annientato da una bomba, più o meno alla linea 300). Dovrete anche stampare il punteggio alla fine del gioco. Nella versione finale, al termine di questa sezione, ho incluso anche il punteggio record del gioco. È abbastanza semplice da aggiungere. Provate ad aggiungere queste piccole modifiche al programma, per renderlo più elegante. Potreste anche provare a definirvi un carattere per il grattacielo per dotarlo di finestre, invece di usare una semplice 'X' inversa.

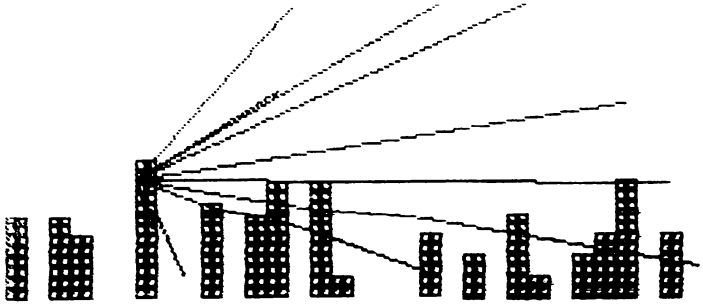
La versione finale contiene delle frasi DATA per definire i caratteri grafici; potete introdurla e farla girare.

Versione finale del programma BOMBARDAMENTO:

```
  2 LET h=0
  4 IF h=0 THEN GO SUB 1000
  5 LET s=0
 10 PAPER 0: INK 6: BORDER 1
 30 PRINT AT 21,0;"Difficolta'
(da 1 a 9) ?": LET d$=INKEY$: IF
 LEN d$<>1 OR CODE d$<49 OR CODE
 d$>57 THEN GO TO 30
 32 CLS
 35 LET d=12-VAL d$
 40 FOR a=0 TO 31
```

```
  50 FOR b=d+INT (RND*d) TO 21
  60 PRINT PAPER 0;AT b,a;CHR$ 1
46
  70 NEXT b
  80 NEXT a
  90 LET u=0
 100 LET p=0
 110 LET f=0
 130 PRINT AT 21,0;CHR$ 20+CHR$
1+d$+" Premi un qualsiasi tasto
per    bombardare"
 140 PRINT AT u,p;: IF PEEK (PEE
K 23684+256*PEEK 23685)=255 THEN
 GO TO 200
 142 PRINT INK 4;CHR$ 144
 144 IF f=0 THEN LET a=u: LET t=
p
 148 IF INKEY$<>"" THEN LET f=1
 155 IF f=1 THEN PRINT AT a,t;"
": LET a=a+1: IF a=22 THEN GO TO
 340
 157 PRINT AT a,t;: IF PEEK (PEE
K 23684+256*PEEK 23685)=255 THEN
 GO TO 300
 158 IF f=1 THEN PRINT  INK 2;CH
R$ 145: BEEP .01,60-a
 160 LET p=p+1
 170 PRINT AT u,p-1;" "
 180 IF p=32 THEN LET p=0: LET u
=u+1: BEEP .1,u
 185 IF u=22 THEN GO TO 1200
 190 GO TO 140
 200 LET x=p*8: LET y=(21-u)*8
 205 FOR a=1 TO s STEP 10
 207 PLOT x,y
 210 DRAW INK 6;INT (RND*256)-x,
INT (RND*158)-y
 220 BEEP .1,20: BEEP .01,10
 230 NEXT a
 235 GO TO 400
 300 FOR a=a TO 21
 310 IF RND>.99 THEN GO TO 340
 315 BEEP .005,a-20
```

```
 320 PRINT AT a,t;" "
 325 LET s=s+1: IF s/250=INT (s/
250) THEN POKE 23624,PEEK 23624+
8
 330 NEXT a
 340 LET f=0: GO TO 140
 400 IF h<s THEN LET h=s
 410 PRINT AT 0,0;"Punteggio: ";
s," record: ";h
 420 GO TO 5
1000 DATA "a",0,0,96,126,127,8,1
6,0,"b",0,0,0,32,28,32,0,0,"c",2
55,153,153,255,255,153,153,255
1005 FOR f=1 TO 3
1007 READ a$
1010 FOR a=0 TO 7
1020 READ b: POKE USR a$+a,b
1030 NEXT a
1045 NEXT f
1050 RETURN
1200 DATA 0,4,7
1210 FOR a=1 TO 3
1215 READ c
1220 FOR b=1 TO 3
1230 BEEP 1/3,c
1240 NEXT b
1250 NEXT a
1255 RESTORE 1200
1260 BEEP 1,12
1270 GO TO 32
```

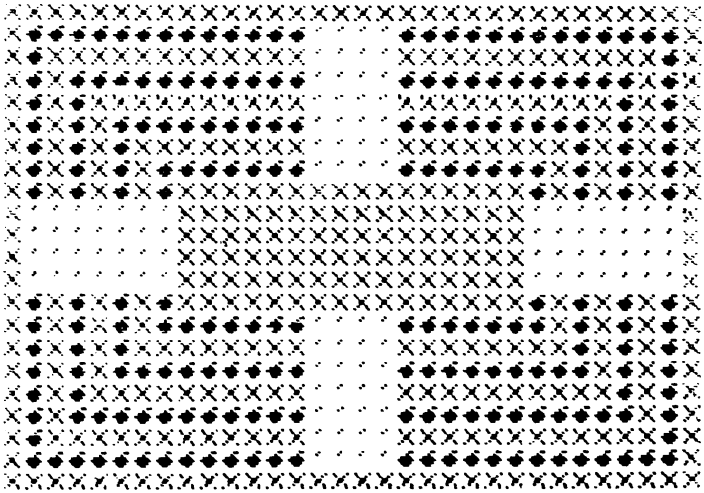


Spero che la spiegazione del gioco vi abbia dato qualche idea su come vanno sviluppati i giochi. Esamineremo altri giochi nel resto del capitolo ma non così in dettaglio come BOMBARDAMENTO.

JACK-MAN

Questo programma è basato in qualche modo su DODGEM (e in parte su PAC-MAN) nel quale il giocatore guida una piccola automobile in un labirinto, segnando punti ogni volta che passa sopra i punti disegnati nel labirinto, e cercando di evitare scontri con la macchina guidata dal computer.

Lo scenario di JACK-MAN è meno macabro. JACK è un codardo ladro di uva che si trova in un vigneto francese, disegnato con molta cura. Lo scopo del gioco è guidare JACK nel vigneto e riuscire a fargli mangiare quanta più uva è possibile. Allo stesso tempo deve eludere l'irato contadino.



Quasi un terzo di tutto il programma è occupato dalla routine che stampa il vigneto con i suoi muri e i grappoli d'uva (l'uva é tutta dello stesso colore e così anche i muri, ma uva e muri hanno colori diversi). Sarebbe saggio registrare la routine seguente, dato che è piuttosto lunga, così potrete aggiungerla in seguito senza doverla reinserire se per qualche ragione vi capitasse di perderla.

Notate che le 'G' devono essere introdotte in modo grafico.

```
  2 LET h=0
  5 LET e$="Jack"
 10 DATA 60,126,240,224,224,240
,126,60
 12 DATA 0,66,195,195,231,255,1
26,60,60,126,15,7,7,15,126,60
 14 DATA 60,126,255,231,195,195
,66,0,0,0,0,0,60,60,60,60
 18 DATA 60,126,255,255,255,255
,126,60,0,6,3,60,126,60,24,0
 20 DATA 24,60,24,255,24,24,36,
102
 25 FOR a=0 TO 7
 30 FOR b=0 TO 7
 40 READ c
 50 POKE USR CHR$ (144+a)+b,c
 60 NEXT b
 70 NEXT a
 75 LET s=0: LET v=1
 77 LET z=4: LET y=0: PAPER 3:
INVERSE 1
 78 LET f=0
 79 CLS
 80 PRINT INK z;"XXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX"
 90 IF f=1 THEN RETURN
100 PRINT INK z;"X"; INK y;"GGG
GGGGGGGGGG'''GGGGGGGGGGGGG"; IN
K z;"X"
110 IF f=1 THEN GO TO 80
120 PRINT INK z;"X"; INK y;"G";
INK z;"XXXXXXXXXXX"; INK y;"''
'"; INK z;"XXXXXXXXXXX"; INK y
;"G"; INK z;"X"
130 IF f=1 THEN GO TO 100
```

```
 140 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"GGGGGGGGGGG''
''GGGGGGGGGGG"; INK z;"X"; INK y
;"G"; INK z;"X"
 150 IF f=1 THEN GO TO 120
 160 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"XX
XXXXXXX"; INK y;"''''"; INK z;"
XXXXXXXXXX"; INK y;"G"; INK z;"X
"; INK y;"G"; INK z;"X"
 170 IF f=1 THEN GO TO 140
 180 PRINT INK z;"x"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"GGGGGGGGG'''GGGGGGGGG"
; INK z;"X"; INK y;"G"; INK z;"X
"; INK y;"G"; INK z;"X"
 190 IF f=1 THEN GO TO 160
 200 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"XXXXXXXX"; I
NK y;"''''"; INK z;"XXXXXXXX"; I
NK y;"G"; INK z;"X"; INK y;"G";
INK z;"X"; INK y;"G"; INK z;"X"
 210 IF f=1 THEN GO TO 180
 220 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"X"; INK y;"G
GGGGGG'''GGGGGGG"; INK z;"X"; I
NK y;"G"; INK z;"X"; INK y;"G";
INK z;"X"; INK y;"G"; INK z;"X"
 230 IF f=1 THEN GO TO 200
 240 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"X"; INK y;"G
"; INK z;"XXXXXXXXXXXXXXX"; INK
 y;"G"; INK z;"X"; INK y;"G"; IN
K z;"X"; INK y;"G"; INK z;"X"; I
NK y;"G"; INK z;"X"
 250 IF f=1 THEN GO TO 220
 260 FOR a=1 TO 4
 270 PRINT INK z;"X"; INK y;"''
''''"; INK z;"XXXXXXXXXXXXXXX";
 INK y;"''''''''"; INK z;"X"
```

```
280 NEXT a
290 LET f=1
300 GO SUB 240
305 INVERSE 0
307 GO SUB 1040
310 LET u=20
315 PAPER 0: INK 6
320 LET p=17
330 LET du=0
335 LET lj=4
```

Dopo aver assegnato i valori iniziali alla variabile che contiene il punteggio record e alle altre (vedi l'elenco completo alla fine del paragrafo) e completato il vigneto, il programma salta ad una routine che:

[i] Stampa il 'super-acino', un acino lampeggiante che vi dà cinque punti invece di uno solo quando lo mangiate; una volta mangiato, dal contadino o da JACK, il super-acino si sposta in un'altra parte del vigneto.

[ii] Suona un motivetto. Molti video-games delle sale giochi suonano un motivetto prima e dopo il gioco che finisce per urtare i nervi. Se vi irrita, potete escludere la parte di routine corrispondente. Lo Spectrum legge le note per la musica dalla frase DATA alla linea 1050.

Il programma salta a questa routine ogni volta che cancella una 'pagina' e la riscrive, il punteggio record viene aggiornato se lo superate.

Durante questa routine molte delle altre variabili usate durante il gioco sono riportate ai loro valori iniziali.

Il programma prosegue con la parte principale che trasferisce il controllo a diverse routine quando il giocatore vuole muovere JACK o quando JACK colpisce un muro.

Le routine sono le seguenti:

a] *Routine di movimento automatico di JACK*

Questa routine cambia la direzione e il carattere che rappresenta JACK quando JACK colpisce un muro; supponiamo ad esempio che egli si stia muovendo verso est e che colpisca un muro, il programma cambia automaticamente la sua direzione in modo che si muova verso nord. Di conseguenza viene modificato anche il carattere che rappresenta JACK, che ora è rivolto verso nord. Il giocatore non può controllare questo procedimento. JACK si muove sempre con un movimento antiorario.

b] *Routine di movimento manuale di JACK*
Ogni filare di uva è circondato da un muro, di conseguenza JACK non può muoversi attraverso i filari. Comunque ci sono quattro aree libere dove JACK può spostarsi in un nuovo filaro e/o evitare l'arrivo del contadino. Questa routine è utilizzata quando JACK si trova in una delle aree libere oppure quando un giocatore sta premendo un tasto. La routine controlla se il giocatore sta muovendo JACK in una direzione consentita e se è così sposta JACK aggiornando le variabili relative, inoltre emette un suono per aiutare l'utente. Se l'utente tenta di muovere JACK illegalmente, viene suonata una nota diversa.

c] *Routine di movimento automatico del contadino*
Questa routine modifica la direzione del contadino quando incontra un muro (vedi la routine di movimento automatico di JACK).

d] *Routine di movimento manuale del contadino*
Questa routine è chiamata quando il contadino si muove in un'area libera oppure quando il contadino non si trova nello stesso filare di JACK. Il contadino cerca sempre di entrare in collisione con JACK. All'inizio però egli si muove soltanto di una posizione; ogni volta che JACK completa una 'pagina' l'agilità del contadino aumenta e il gioco diventa più difficile.

e] *Routine di pulizia della pagina*
Viene usata quando JACK ha mangiato tutta l'uva e deve essere costruito un nuovo vigneto. Non è una vera e propria routine ma piuttosto una serie di comandi che richiamano parti di altre routine.

f] *Routine di fine gioco*
Questa routine viene chiamata quando il contadino raggiunge JACK. JACK si accartoccia mentre vengono prodotti degli effetti sonori adeguati; se il giocatore ha superato il punteggio record quest'ultimo viene aggiornato e il programma chiede il nome del giocatore. L'azione ritorna all'inizio (tuttavia il punteggio record viene conservato).

g] *Routine del super-acino*
Questa routine viene chiamata all'inizio del programma e ogni volta che JACK mangia un super-acino. Essa cancella il vecchio super-acino (resettando i$), poi sceglie un punto casuale dello schermo e controlla se è una posizione adeguata per introdurvi il nuovo super-acino. Se non è così viene scelto un altro punto casuale. La posizione adatta per il super-acino è quella che verifica le seguenti condizioni:

i Un punto che non si trova su un muro
ii Un punto che non si trova in una delle aree libere
iii Un punto che si trova in un filare
iv Un punto che si trova in una posizione dove JACK ha mangiato un acino quando sono stati mangiati più della metà degli acini.

Il programma principale muove JACK e il contadino nelle rispettive direzioni e richiama le routine descritte prima quando è necessario. Le routine si trovano alle seguenti linee:

a]	Routine di movimento automatico di JACK	1000-1015
b]	Routine di movimento manuale di JACK	565-700
c]	Routine di movimento automatico del contadino	1020-1030
d]	Routine movimento manuale del contadino	710-790
e]	Routine di fine gioco	800-900
f]	Routine del super-acino	1150-1250

Ho lavorato a turno su ogni routine e poi ho perfezionato il gioco intero.

Potreste trovare più facile introdurre il programma con l'aiuto di un amico che vi legga il listato (specialmente la routine che stampa il vigneto con tutte le INK e PAPER usate).

Quando inserite il programma è molto importante usare la 'G' grafica e non quella normale, o l'uva non verrà stampata. Potrebbe divertirvi introdurre solo la parte del programma che definisce i caratteri per vedere come appare l'uva quando premete la 'G' grafica.

Listato finale di JACK-MAN:

```
    2 LET h=0
    5 LET e$="Jack"
   10 DATA 60,126,240,224,224,240
,126,60
   12 DATA 0,66,195,195,231,255,1
26,60,60,126,15,7,7,15,126,60
   14 DATA 60,126,255,231,195,195
,66,0,0,0,0,0,60,60,60,60
   18 DATA 60,126,255,255,255,255
,126,60,0,6,3,60,126,60,24,0
   20 DATA 24,60,24,255,24,24,36,
102
   25 FOR a=0 TO 7
   30 FOR b=0 TO 7
   40 READ c
   50 POKE USR CHR$ (144+a)+b,c
   60 NEXT b
   70 NEXT a
   75 LET s=0: LET v=1
   77 LET z=4: LET y=0: PAPER 3:
```

```
INVERSE 1
  78 LET f=0
  79 CLS
  80 PRINT INK z;"XXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX"
  90 IF f=1 THEN RETURN
 100 PRINT INK z;"X"; INK y;"GGG
GGGGGGGGGG'''GGGGGGGGGGGGG"; IN
K z;"X"
 110 IF f=1 THEN GO TO 80
 120 PRINT INK z;"X"; INK y;"G";
 INK z;"XXXXXXXXXXXX"; INK y;"''
''"; INK z;"XXXXXXXXXXXX"; INK y
;"G"; INK z;"X"
 130 IF f=1 THEN GO TO 100
 140 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"GGGGGGGGGGG''
''GGGGGGGGGGG"; INK z;"X"; INK y
;"G"; INK z;"X"
 150 IF f=1 THEN GO TO 120
 160 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"XX
XXXXXXXX"; INK y;"''''"; INK z;"
XXXXXXXXX"; INK y;"G"; INK z;"X
"; INK y;"G"; INK z;"X"
 170 IF f=1 THEN GO TO 140
 180 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"GGGGGGGGG'''GGGGGGGGG"
; INK z;"X"; INK y;"G"; INK z;"X
"; INK y;"G"; INK z;"X"
 190 IF f=1 THEN GO TO 160
 200 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"XXXXXXXX"; I
NK y;"''''"; INK z;"XXXXXXXX"; I
NK y;"G"; INK z;"X"; INK y;"G";
INK z;"X"; INK y;"G"; INK z;"X"
 210 IF f=1 THEN GO TO 180
 220 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"X"; INK y;"G
GGGGGG'''GGGGGGG"; INK z;"X"; I
```

```
NK y;"G"; INK z;"X"; INK y;"G";
INK z;"X"; INK y;"G"; INK z;"X"
 230 IF f=1 THEN GO TO 200
 240 PRINT INK z;"X"; INK y;"G";
 INK z;"X"; INK y;"G"; INK z;"X"
; INK y;"G"; INK z;"X"; INK y;"G
"; INK z;"XXXXXXXXXXXXXXX"; INK
 y;"G"; INK z;"X"; INK y;"G"; IN
K z;"X"; INK y;"G"; INK z;"X"; I
NK y;"G"; INK z;"X"
 250 IF f=1 THEN GO TO 220
 260 FOR a=1 TO 4
 270 PRINT INK z;"X"; INK y;"'''
''''"; INK z;"XXXXXXXXXXXXXXX";
 INK y;"''''''''"; INK z;"X"
 280 NEXT a
 290 LET f=1
 300 GO SUB 240
 305 INVERSE 0
 307 GO SUB 1040
 310 LET u=20
 315 PAPER 0: INK 6
 320 LET p=17
 330 LET du=0
 335 LET lj=4
 340 LET dp=1
 345 LET lf=4
 350 LET a=10
 355 LET g$="'"
 360 LET t=1
 365 LET l$="'"
 370 LET da=-1
 375 PRINT AT 9,9; PAPER 0; INK
4; INVERSE 1;"Punti: "
 380 LET dt=0
 385 LET q=1
 390 LET c=144
 400 IF INKEY$<>"" THEN BEEP .01
,0
 410 PRINT AT u,p; INK 6;CHR$ c
 440 IF SCREEN$ (u+du,p+dp)="X"
THEN GO TO 1000
 460 PRINT AT a,t; INK 3;g$
```

```
 470 IF SCREEN$ (a+da,t+dt)="X"
THEN GO TO 1020
 490 LET g$=SCREEN$ (a+da,t+dt)
 495 IF ATTR (a+da,t+dt)>128 THE
N LET g$=","
 500 IF g$<>"'" THEN LET q=v
 510 IF g$<>"'" AND g$<>"," THEN
 LET g$="G"
 515 LET a=a+da: LET t=t+dt
 517 IF a=ba AND t=rl THEN GO SU
B 1200
 520 IF SCREEN$ (a,t)="'" AND lj
<>lf THEN GO SUB 710
 525 PRINT AT a,t;"H"
 530 PRINT AT u,p; INK 3;l$
 540 LET u=u+du: LET p=p+dp
 542 IF ATTR (u,p)=6 THEN GO TO
800
 545 LET m=0: LET l$=SCREEN$ (u,
p): IF l$="" THEN LET l$=",": LE
T m=1
 547 IF u=ba AND p=rl THEN GO SU
B 1150
 550 PRINT AT u,p;"F"
 555 IF m=1 THEN LET s=s+1: LET
s2=s2+1: BEEP .005,-10: BEEP .00
5,-5: PRINT AT 9,17;s
 557 IF s2>=224 THEN GO TO 1110
 560 IF INKEY$="" OR l$="," THEN
 GO TO 400
 565 LET i$=INKEY$
 570 IF CODE INKEY$<53 OR CODE I
NKEY$>56 THEN GO TO 400
 580 RESTORE 585
 585 DATA du,"6","7",dp,"5","8"
 590 FOR i=1 TO 2
 600 READ j: READ j$: READ k$
 610 IF j=0 AND INKEY$<>J$ AND I
NKEY$<>k$ THEN GO TO 400
 620 NEXT i
 630 LET u1=u: LET p1=p
 640 LET u1=u+((i$="6")-(i$="7")
)*(du=0)*2
```

```
 650 LET p1=p+((i$="8")-(i$="5")
)*(dp=0)*2
 655 LET n=(i$="6")*(dp=1)+(i$="
7")*(dp=-1)+(i$="5")*(du=1)+(i$=
"8")*(du=-1)
 660 IF n=0 THEN LET n=-1
 665 IF lj+n=0 OR lj+n=5 THEN GO
 TO 400
 670 LET lj=lj+n
 675 PRINT AT u,p; INK 3;l$
 680 LET u=u1: LET p=p1
 690 PRINT AT u,p;"F"
 695 BEEP .01,10
 700 GO TO 400
 710 IF INT q=0 THEN RETURN
 715 LET tf=lf+(lf<lj)-(lf>lj)
 720 LET q=q-1
 730 LET o=(lf>lj)*(da=-1)+(lf<l
j)*(da=1)
 740 IF o=0 THEN LET o=-1*(dt=0)
 745 LET o=o*2
 750 LET t=t+o
 760 LET o=(lf>lj)*(dt=1)+(lf<lj
)*(dt=-1)
 765 LET lf=tf
 770 IF o=0 THEN LET o=-1*(da=0)
 775 LET o=o*2
 780 LET a=a+o
 790 RETURN
 800 RESTORE 900
 805 FOR a=1 TO 4
 810 READ b$: READ r
 820 PRINT AT u,p;b$
 830 BEEP 1,r
 840 NEXT a
 850 FOR a=1 TO 64
 860 BEEP .01,25
 870 NEXT a
 872 PRINT AT 17,11;"GAME OVER"
 875 IF h<s THEN INPUT "Hai stab
ilito un nuovo record:  introduc
i il tuo nome e premi    ENTER.
";e$: IF LEN e$>11 THEN PRINT AT
```

```
 u,p-5;"troppo lungo.....":
GO TO 800
 880 IF h<s THEN LET h=s
 885 IF INKEY$="" THEN GO TO 885
 890 GO TO 75
 900 DATA "F",30,"E",20,".",10,"
 ",0
1000 IF du=0 THEN LET du=-dp: LE
T dp=0: LET c=c+1: GO TO 420
1010 IF dp=0 THEN LET dp=du: LET
 du=0: LET c=c+1: IF c=148 THEN
LET c=144
1015 GO TO 420
1020 IF da=0 THEN LET da=dt: LET
 dt=0: GO TO 515
1030 IF dt=0 THEN LET dt=-da: LE
T da=0: GO TO 515
1040 RESTORE 1050
1042 LET s2=0
1045 GO SUB 1200
1050 DATA 7,1,7,.5,7,.5,10,1,12,
1,14,.5,12,1.5,10,1.5,12,.5,7,1,
7,.5,7,.5,10,1,12,1,7,2
1055 DATA 7,1,7,1,10,1,12,1,14,.
5,12,1.5,10,1,12,1,7,1,7,1,5,.5,
4,1.5,0,2
1060 FOR a=1 TO 28
1070 READ w: READ x
1080 BEEP x/10,w
1090 NEXT a
1095 IF h<s THEN LET h=s
1100 PRINT AT 11,9; PAPER 4; INK
 0;"Record: ";h;AT 12,9;"di ";e$
1105 RETURN
1110 LET v=v+RND
1120 GO TO 77
1150 FOR d=24 TO 0 STEP -1
1155 BEEP .01,d
1160 NEXT d
1165 BEEP .1,36
1170 LET s=s+5
1180 LET l$=","
1200 LET ba=INT (RND*20)+1
```

```
1210 LET rl=INT (RND*29)+1
1220 LET b$=SCREEN$ (ba,rl)
1230 IF b$="'" OR b$="X" OR (b$=
"," AND s2<112) OR (ba>8 AND ba<
13) THEN GO TO 1200
1240 PRINT AT ba,rl; FLASH 1; IN
K 5; PAPER 0;"G"
1245 IF s>=112 THEN LET s2=s2+1
1250 RETURN
```

Variabili usate in JACK-MAN:

h	punteggio record fino a questo momento
e$	nome del giocatore che ha stabilito il punteggio record
a	variabile di controllo per diversi cicli FOR/NEXT
b	variabile di controllo per diversi cicli FOR/NEXT
c	variabile usata per leggere i dati dei caratteri grafici
s	punteggio
v	numero di linee che il contadino può saltare nelle aree libere
z	colore dei muri
y	colore dell'uva
f	flag per la routine di stampa
u	coordinata y di JACK
p	coordinata x di JACK
du	direzione nord o sud di JACK (può essere –1 o 1)
lj	la linea del filare in cui si trova JACK (1-4))
dp	direzione est o ovest di JACK (vedi du)
lf	la linea del filare in cui si trova il contadino (vedi lj)
a	coordinata y del contadino
g$	carattere che cancella il contadino
t	coordinata x del contadino
l$	carattere che cancella JACK
da	direzione nord o sud del contadino (vedi du)
dt	direzione est o ovest del contadino (vedi dp)
q	il numero di linee che il contadino può saltare in questa particolare area
c	carattere usato per stampare JACK
ba	coordinata y del super-acino
rl	coordinata x del super-acino
m	flag per indicare se un acino è stato mangiato
i	variabile di controllo per i cicli FOR/NEXT
j	legge le direzioni consentite nella routine di movimento manuale di JACK
j$, k$	legge i caratteri ammessi nella routine di movimento manuale di JACK (come j$)

u1, p1 — valore temporaneo della coordinata di JACK mentre la routine di movimento manuale controlla se il movimento richiesto è ammesso

p1 — come u1

n — modifica la variabile che contiene la linea in cui si trova JACK

o — direzione in cui si deve muovere il contadino per allinearsi con JACK

b$ — legge i caratteri da stampare per accartocciare JACK

r — legge le note da suonare quando Jack viene accartocciato

s2 — numero di acini mangiati fino a questo momento nella pagina attuale

w — legge le note da suonare nel motivetto di apertura

x — legge la durata delle note del motivetto di apertura

i$ — tasto premuto nella routine di movimento manuale di JACK

Le variabili che leggono i dati tramite l'istruzione READ, sono quelle situate nei cicli FOR/NEXT e di conseguenza cambiano il loro contenuto continuamente.

POESIA

I primi due giochi che abbiamo esaminato in questo capitolo erano programmi grafici di azione; in essi l'interazione con il giocatore consiste nel prendere decisioni in accordo con i tasti premuti, per modificare la posizione degli oggetti sullo schermo.

L'interazione con l'utente, nel prossimo programma, è nulla; l'unica eccezione consiste nel sedersi e ammirare i risultati prodotti dal programma. Il programma è costituito in modo da poter essere inserito e fatto girare così come è. Poi inizia la parte più interessante, in cui aggiungerete le vostre parole al vocabolario contenuto nelle frasi DATA dalla linea 200 alla 320.

```
  10 REM POESIA
  20 DIM A$(13,12): DIM B(13)
  30 RANDOMIZE : GO TO 90
  40 FOR M=1 TO 13
  50 RESTORE 190+10*M
  60 FOR G=1 TO RND*10+1:  READ
B$: NEXT G: LET A$(M)=B$
  70 LET B(M)=LEN B$: NEXT M
  80 RETURN
  90 POKE 23692,0
 100 FOR H=1 TO 10
 110 GO SUB 40
 120 PRINT '"UN";A$(1)( TO B(1))
;" ";A$(2)( TO B(2));" ";A$(3)(
TO B(3)),"   NEL";A$(4)( TO B(4)
);","
```

```
 130 PRINT A$(5)( TO B(5));" UN"
;A$(6)( TO B(6));"  PER ";A$(7)(
TO B(7))," ";A$(8)( TO B(8));"..
."
 140 PRINT  A$(9)( TO B(9));" ";
A$(10)( TO B(10));" E ";A$(11)(
TO B(11))
 150 PRINT TAB 5;"...";A$(12)( T
O B(12));", ";A$(13)( TO B(13));
"."
 160 PAUSE 200
 170 INK RND*6
 180 NEXT H
 190 STOP
 200 DATA " UOMO"," RAGAZZO","A
DONNA"," BAMBINO"," OMBRA"," ECO
","A RAGAZZA","O STALLONE"," PRO
FETA"," MARTIRE"
 210 DATA "CANTAVA","ASPETTAVA",
"PARLAVA","STRILLAVA","SI DIFEND
EVA","SOSTAVA","CADEVA","INCIAMP
O'"
 220 DATA "TRISTEMENTE","ALLEGRA
MENTE","LENTAMENTE","LANGUIDAMEN
TE","PAZZAMENTE","UMILMENTE","FO
RTEMENTE","DOLCEMENTE","CHETAMEN
TE","VELOCEMENTE"
 230 DATA " BUIO","L'ANTICAMERA"
,"LA STANZA","LA MATTINA","LA SE
RA"," GIARDINO","LA STRADA"," BO
SCO SCURO"," CANYON","LA FORESTA
"
 240 DATA "GUARDANDO","CERCANDO"
,"SOSPIRANDO","TOCCANDO","RAGGIU

NGENDO","FISSANDO","CHIEDENDO","
DIFENDENDO","ASPETTANDO","PIANGE
NDO"
 250 DATA " MODO"," SENTIERO","
PASSO"," SEGNO"," DESIDERIO"," B
ISOGNO","A TORCIA","A PORTA"," C
ANCELLO","A MORTE"
```

```
 260 DATA "RAGGIUNGERE","TOCCARE
","MUOVERE","RUBARE","DIVIDERE",
"RUOTARE","INTRECCIARE","AFFRONT
ARE","SPINGERE","BIASIMARE"
 270 DATA "LA LUNA","IL MARE","L
A NOTTE","L'AMORE","L'ODIO","IL
SOLE","LA PAURA","LA GIOIA","LA
LUCE","IL GIORNO","IL DOLORE"
 280 DATA "PORTANDO","GUARDANDO"
,"PASSANDO","RIMPROVERANDO","TEN
TANDO","AGITANDOSI","TREMANDO","
MERAVIGLIANDOSI","INSEGNANDO","P
ARLANDO"
 290 DATA "IL CUORE","PER COME",
"PERCHE'","GIU'","ATTRAVERSO","P
ER","QUINDI","APPENA","COME","OR
A"
 300 DATA "NIENTE","DANDO","SCOM
PARENDO","OSANDO","PARTENDO","RI
DENDO","PIANGENDO","SGRIDANDO","
CURANDO","AFFERRANDO"
 310 DATA "PORTANDO","GUARDANDO"
,"PASSANDO","RIMPROVERANDO","TEN
TANDO","AGITANDOSI","TREMANDO","
MERAVIGLIANDOSI","INSEGNANDO","P
ARLANDO"
 320 DATA "NIENTE","DANDO","SCOM
PARENDO","OSANDO","PARTENDO","RI
DENDO","PIANGENDO","SGRIDANDO","
CURANDO","AFFERRANDO"
```

È abbastanza semplice scrivere programmi generatori di poesie per il computer, se seguite una semplice procedura. Il trucco consiste nello scrivere un verso di poesia per proprio conto, ricavarne le parole usate e passarle al computer che le selezionerà a caso per riempire spazi prefissati.

Questo particolare programma é stato scritto partendo dal seguente brano poetico, non molto brillante:

l'AQUILA VOLAVA RAPIDAMENTE
nelle NUVOLE,
CERCANDO un MODO
per TOCCARE IL CIELO...
VOLANDO POI e GRIDANDO ... PRECIPITANDO, MUOVENDOSI

Poi ho compilato una lista di parole che avrebbero dovuto sostituire le parole in maiuscolo. Queste parole si trovano nelle frasi DATA a partire dalla linea 200n

Una volta che questo è stato fatto, la costruzione vera e propria del programma é molto semplice. La linea 20 dimensiona due matrici, una per memorizzare la parola selezionata e l'altra per memorizzare la lunghezza di quella parola. Poichè ci sono tredici parole da inserire in ogni verso, viene usato un ciclo da 1 a 13 (linea 40). La parola RESTORE dice al computer di posizionarsi su un particolare gruppo di dati primi di leggerli con l'istruzione READ. Lo Spectrum vi consente di usare una RESTORE selettiva, cioè una RESTORE ad una particolare linea di programma. La linea 50 esegue una RESTORE alla linea 190 + 10 x M. Questo assicura che la linea 60, che seleziona casualmente una delle parole contenute in una frase DATA, inizi sempre il suo conteggio dalla prima parola di un gruppo di datin La linea 70 memorizza la lunghezza della parola nella matrice B.

La linea 90, dove inizia realmente il programma, assegna alla locazione di memoria 23692 il valore 0. Questo fa si che quando lo schermo è riempito, lo scroll entri automaticamente in funzione, senza porre la domanda 'scroll?'. La linea 100 controlla il ciclo di conteggio dei versi scritti dal computer. Con la versione attuale del programma, lo Spectrum scriverà dieci versi, ma naturalmente non c'é ragione perchè non dobbiate modificare questo valore se desiderate un numero di versi maggiore o minore.

Abbiamo già parlato della routine che inizia alla linea 40, chiamata dalla linea 110. Essa seleziona le parole usate nel verso, carica queste parole negli elementi della matrice A$ e la loro lunghezza nella matrice B.

Le linee dalla 120 alla 150 stampano il poeman Abbiamo bisogno di qualcosa di molto poco elegante come (TO B (1)) dopo ogni elemento della matrice di stringhe per evitare che vengano stampati anche gli spazi eccedenti (sullo Spectrum le stringhe di una matrice sono sempre completate con spazi se risultano più corte della lunghezza dichiarata). Senza questo accorgimento il poema risulterebbe veramente molto strano sullo schermo.

Dopo che un verso è stato stampato il computer effettua una pausa (di circa quattro secondi), modifica casualmente il colore del testo (linea 170) e prosegue con la stampa del prossimo verso.

Come ho detto all'inizio della descrizione di questo programma, la parte veramente divertente viene quando sostituite le parole contenute nelle frasi DATA con le vostren Una volta fatto ciò potreste voler riscrivere l'intero programma partendo da un brano di poesia completamente diverso.

METEORE

In questo gioco siete il pilota di un'astronave da esplorazione che attraversa lo spazio; improvvisamente vi trovate in uno sciame di meteoriti. Dovete cercare di evitarle il più a lungo possibile. Fortunatamente, all'interno dello sciame si trova an-

che un certo numero di scorte di carburante che vi serviranno per rifornire la vostra astronave, semplicemente passandoci attraverso (non è meravigliosa la tecnologia moderna?).

Ricorderete che nel programma POESIA (linea 90) abbiamo azzerato la locazione di memoria 23692 per impedire allo Spectrum di fermarsi dopo aver stampato 22 linee con il messaggio 'scroll?'. Faremo lo stesso in questo programma, dove vi trovate al centro dello schermo e vedete l'universo muoversi attorno a voi. Il sistema più semplice per ottenere che il computer faccia scorrere lo schermo verso l'alto in modo continuo è stampare un carattere NEWLINE in fono allo schermo; in questo modo il testo verrà fatto slittare verso l'alto per far posto alla nuova linea. Il carattere NEWLINE ha il codice 13 (CHR$ 13).

Usate i caratteri grafici descritti nelle frasi REM per rappresentare i vari simboli. Sebbene non possiate vedere i simboli apparire immediatamente, le lettere grafiche saranno convertite nei nuovi caratteri grafici non appena verranno eseguite le linee 5 - 70. Ancora una volta potete inserire prima queste linee e farle eseguire per veder comparire un meteorite ogni volta che premete la B grafica.

Il carburante é memorizzato come una stringa; esso viene decrementato di un punto ogni volta che premete un tasto e contemporaneamente l'astronave è spinta verso destra (essa scivola automaticamente verso sinistra se non premete nessun tasto). L'Iperspazio (che muove l'astronave in un punto casuale dello schermo) consuma molto carburante; d'altra parte esso va usato solo nei casi di emergenza; perchè potreste materializzarvi su un meteorite. Se il carburante comincia a scarseggiare, potete lasciare che l'astronave vada lentamente alla deriva (in questo modo non consumate), ma attenzione! Se arrivate al lato sinistro dello schermo perdete punti invece di guadagnarli. Vi potete rifornire toccando i distributori di benzina blu. Se terminate il carburante o colpite un meteorite l'astronave salterà in aria. L'esplosione é generata nello stesso modo di BOMBARDAMENTO. Le meteore diventano più grandi ogni 500 punti. Come in BOMBARDAMENTO più è alto il punteggio e più è violenta l'esplosione. Entrate in iperspazio premendo lo spazio, qualunque altro tasto vi consente di spostarvi verso destra.

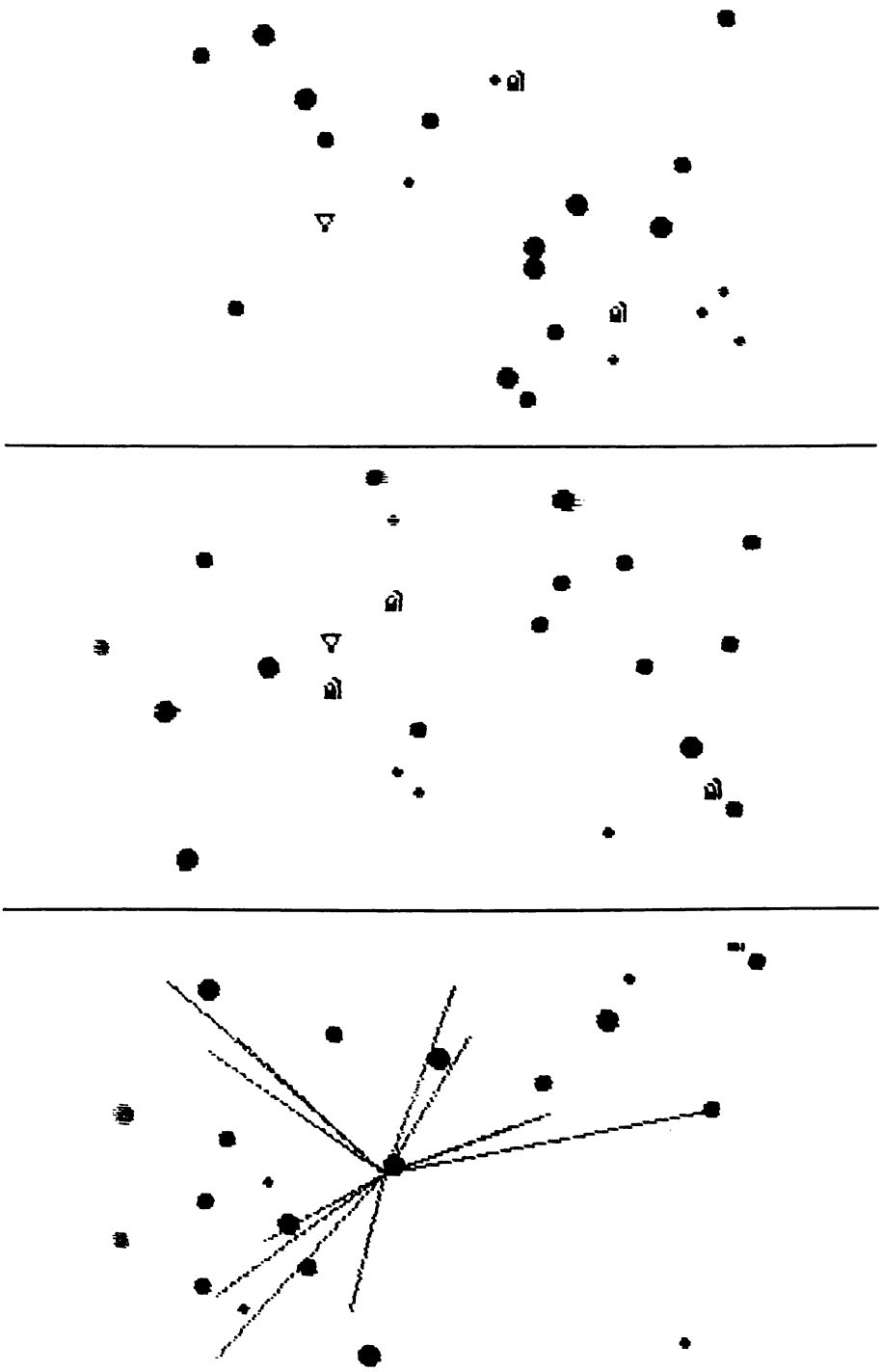
Potete provare a migliorare il gioco aggiungendo le routine necessarie per sparare (magari contro astronavi nemiche che farete comparire di tanto in tanto).

Listato del programma METEORE:

```
  5 LET h=0
 10 LET u=11
 20 LET p=16
 22 LET s=0
 25 IF h>0 THEN GO TO 80
 30 DATA "a",255,66,66,36,24,24
,24,0,"b",60,126,255,255,255,255
,126,60
```

```
  32 DATA "c",0,60,126,126,126,1
26,60,0,"d",0,0,24,60,60,24,0,0
  33 DATA "m",12,6,50,74,74,122,
122,122
  35 FOR b=1 TO 5
  37 READ a$
  40 FOR a=0 TO 7
  50 READ c: POKE USR a$+a,c
  60 NEXT a
  70 NEXT b
  72 DATA 255,126,63,31,15,7,3,1
  74 FOR a=0 TO 7: READ c
  76 FOR b=0 TO 7: POKE USR CHR$
 (a+101)+b,c
  78 NEXT b: NEXT a
  80 BORDER 0: PAPER 0: INK 6: C
LS
  82 LET a$="                "
  84 LET sl=16
  86 LET sa=148
  90 REM Al posto di <A> , <B> ,
 100 REM <C> , <D> , <M> usare i
 110 REM corrispondenti tasti
 120 REM grafici
 130 IF SCREEN$ (u,p)<>" " THEN
GO TO 200
 132 PRINT INK 4;AT u,p;"<A>"
 135 POKE 23692,255
 137 LET s=s+1: PRINT AT 0,0;s
 139 PRINT AT 0,7;"Energia: "; I
NK (sl>=25)*-1+6;TAB sl;a$
 140 FOR a=1 TO s/500+1: PRINT I
NK 2;AT 21,INT (RND*32);CHR$ (14
5+INT (RND*3)): NEXT a
 141 IF RND>.95 THEN PRINT INK 5
;AT 21,INT (RND*31);"<M>"
 142 IF INKEY$<>"" THEN LET sa=s
a+1: IF sa>=156 THEN LET sl=sl+1
: LET a$=a$(2 TO  ): LET sa=148
 143 IF sl=31 AND sa=148 THEN GO
 TO 200
 144 LET a$(1)=CHR$ sa
 147 IF sl>=25 THEN BEEP .007,sl
```

```
 150 PRINT AT u,p;" "
 155 IF INKEY$=" " THEN LET p=IN
T (RND*32): IF LEN a$<=1 THEN GO
 TO 200: LET a$=a$(2 TO ): LET s
l=sl+1
 160 PRINT AT 21,0;CHR$ 13
 170 LET p=p-(p>0)+(INKEY$<>"")*
2*(p<31)
 175 IF p=0 THEN LET s=s-2*(s>0)
: BEEP .01,s/20
 180 GO TO 130
 200 IF ATTR (u,p)=5 THEN LET a$
="[redacted]": LET sl=16: P
RINT AT u,p;"*": FOR a=-sl TO 30
: BEEP .01,a: NEXT a: GO TO 132
 202 LET x=p*8: LET Y=(21-u)*8
 205 INK 3
 207 FOR c=1 TO s/10
 210 PLOT x,y
 220 LET a=INT (RND*256): LET b=
INT (RND*158)
 230 DRAW a-x,b-y
 235 IF RND>.85 THEN BEEP .01,20
 240 NEXT c
 250 INK 6
 260 BEEP 1,20
 265 IF h<s THEN LET h=s
 270 PRINT AT 21,0;"Punti: ";s,"
Record: ";h
 280 PRINT AT 0,0;"Premi un tast
o qualsiasi"
 290 IF INKEY$="" THEN GO TO 290
 300 GO TO 10
```



CAMPO DI TESCHI

Con un nome così terrificante é probabile che non pensiate di trovarvi davanti a un gioco veramente piacevole. Giocare è semplice, ma vincere è quasi impossibile.

Siete perseguitato da dieci teschi lampeggianti. Essi sanno dove vi trovate e impiegano gran parte del tempo nel tentativo di raggiungervi. Tuttavia i teschi sono piuttosto tonti e possono essere 'uccisi' se riuscite ad attirarli nei rettangoli colorati che sono sparsi per lo schermo. Il solo modo in cui potete farlo é di raggiungere un lato del rettangolo mentre il teschio si trova sul lato opposto. Siccome il teschio vi insegue, andà a urtare ciecamente contro il rettangolo ('ha occhi solo per voi') e svanirà con un allegro BEEP.

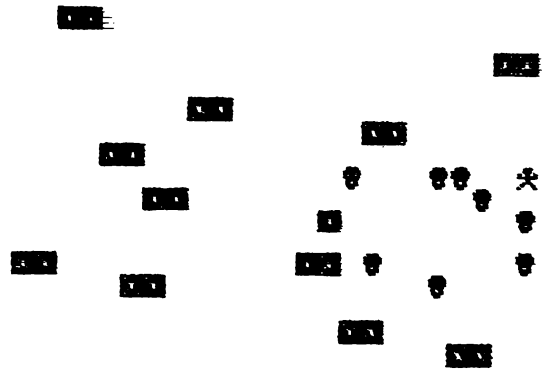
Potete muovervi usando i tasti "5", "6", "U", e "8", per spostarvi nelle direzioni indicate dalla frecce sui tasti. Non potete attraversare i rettangoli colorati, ma se finite contro uno di essi non rimanete feriti. La sola cosa che può ferirvi é un teschio. Il gioco ha termine quando uno dei teschi vi raggiunge, oppure quando siete riusciti ad attirare tutti i teschi nella 'morte rettangolare'n

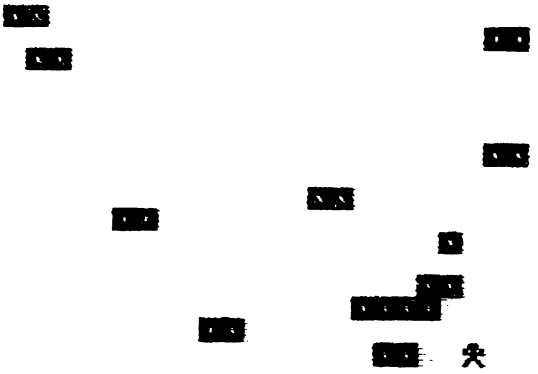
Ricorderete certamente come sono stati definiti i caratteri grafici negli altri giochi di questo capitolo. Avere un ciclo separato per ogni carattere da definire può essere piuttosto dispendioso; con un unico ciclo si possono definire più caratteri diversi, ma il procedimento diventa un po' più complesso. Il ciclo dalla linea 400 alla 430 definisce sia il carattere per l'uomo che quello per il teschio. La frase DATA contiene i dati per entrambi, mischiati assieme. La "A" grafica diventa l'uomo e la "B" diventa il teschio; questo significa che dovete inserire una "B" grafica alla fine della linea 70 e una "A" grafica alla fine delle linee 140 e 270.

```
  10 REM CAMPO DI TESCHI
  20 GO SUB 310: REM VARIABILI
  30 FOR B=1 TO 10
  40 IF A(B,1)=1 THEN GO SUB 220
  50 PRINT AT C(B,2),C(B,3);" "
  60 IF A(B,1)<>1 THEN GO TO 180
  70 PRINT AT A(B,2),A(B,3); INK
 B/2; FLASH 1;"<B>"
  80 LET C(B,2)=A(B,2): LET C(B,
3)=A(B,3)
  90 LET N=N-(INKEY$="5" AND N>0
)+(INKEY$="8" AND N<31)
 100 IF A(B,2)>M THEN LET A(B,2)
=A(B,2)-1
 110 LET M=M+(INKEY$="6" AND M<1
9)-(INKEY$="7" AND M>0)
 120 IF A(B,2)<M THEN LET A(B,2)
```

```
=A(B,2)+1
 130 IF SCREEN$ (M,N)="X" THEN L
ET M=X: LET N=Y
 140 PRINT AT X,Y;" ";AT M,N;"<A
>"
 150 IF A(B,3)<N THEN IF A(B,3)<
28 THEN LET A(B,3)=A(B,3)+INT (4
*RND)
 160 LET X=M: LET Y=N
 170 IF A(B,3)>N THEN IF A(B,3)>
3 THEN LET A(B,3)=A(B,3)-INT (4*
RND)
 180 NEXT B
 190 IF SCORE<10 THEN GO TO 30
 200 PRINT AT 2,0; FLASH 1; BRIG
HT 1; INK 2; PAPER 6;"HAI BATTUT
O I TESCHI!!!!!"
 210 BEEP .008,60-120*RND: GO TO
 210
 220 REM CONTROLLO TESCHI
 230 IF SCREEN$ (A(B,2),A(B,3))=
"X" THEN LET A(B,1)=0: FOR H=1 T
O 10: BEEP .01,5*H: BEEP .01,50-
5*H: NEXT H: LET SCORE=SCORE+1:
PRINT AT 2,22; FLASH 1; BRIGHT 1
; INK 4;"PUNTI> ";SCORE
 240 IF A(B,2)=M AND A(B,3)=N TH
EN PRINT AT C(B,2),A(B,3);" ": G
O TO 260
 250 RETURN
 260 REM FINE DEL GIOCO
 270 PRINT AT A(B,2),A(B,3); FLA
SH 1; BRIGHT 1; INK 2;"<A>"
 280 BEEP .01,RND*20+40
 290 GO TO 280
 300 STOP
 310 REM INIZIALIZZAZIONE
 320 LET SCORE=0
 330 DIM A(10,3): DIM C(10,3)
 340 FOR B=1 TO 10
 350 LET A(B,1)=1
 360 LET A(B,2)=10+INT (RND*9)
 370 LET A(B,3)=10+INT (RND*19)
```

```
 380 LET C(B,2)=A(B,2): LET C(B,
3)=A(B,3)
 390 NEXT B
 400 FOR J=0 TO 7
 410 READ A: READ B
 420 POKE USR "A"+J,A: POKE USR
"B"+J,B
 430 NEXT J
```





```
 440 DATA 28,60,28,126,73,90,127
,126,8,60,20,35,34,60,65,24
 450 LET M=INT (RND*3)+9
 460 LET N=30-INT (RND*30)
 470 LET X=M: LET Y=N
 480 PAPER 7: BORDER 7: CLS
 490 INVERSE 1
 500 FOR G=1 TO 12
 510 PRINT INK G/2;AT RND*16+4,R
ND*26+4;"XX"
 520 NEXT G
 530 INVERSE 0
 540 RETURN
```

BREAKOUT

Lo scopo di BREAKOUT é distruggere i mattoni disegnati nella parte alta dello schermo per mezzo di una pallina; potete far rimbalzare la pallina tramite una racchetta che compare nella parte bassa dello schermo e che potete muovere a destra e a sinistra. Se mancate la palla il gioco finisce.

Il gioco si compone in cinque parti:

I stampare i mattoni che devono essere distrutti
II muovere la racchetta quando il giocatore lo richiede
III far rimbalzare la pallina quando colpisce il bordo dello schermo
IV segnare punti per ogni mattone distrutto
V far terminare il gioco quando l'utente perde la palla

La prima parte è semplice e non richiede spiegazioni. La seconda parte deve identificare il tasto premuto e muovere la racchetta di conseguena. Controlla anche che la racchetta non esca dai bordi dello schermo.

Tutto questo viene realizzato con una sola linea di programma, sfruttando le capacità logiche dello Spectrum che assegna il valore uno ad una frase vera e il valore zero ad una frase falsa. Ad esempio: l'espressione (INKEY$="5") è uguale a uno se è stato premuto il tasto "5", uguale a zero in caso contrario.

Ora, se 'p' è la coordinata orizzontale della racchetta, possiamo usare la seguente espressione:

```
LET p=p+((INKEY$="8")*(p<26)-(INKEY$="5")*(p>0))*2
```

Quando INKEY$ è uguale a "5", l'espressione (INKEY$="8") vale zero e niente è aggiunto a 'p' (infatti p × 0 * 1 = p + 0 = p); ma quando 'p' diventa maggiore di zero (p >0) = 1, così 1 × 1 × 2 = 2 viene sottratto a 'p'.

Questo è un valido sistema per muovere degli oggetti orizzontalmente sullo schermo. Ricordate che un'espressione fra parentesi del tipo (a+b=c) assume il valore 1 se vera, 0 se falsa; così (6+6=12) è uguale a uno ma (9*2=81) è uguale a zero.

La terza parte, che fa rimbalzare la palla contro il muro, cambia soltanto il segno della variabile che contiene la 'direzione' della pallina. Se la pallina ha una direzione +1 che la dirige verso destra e colpisce un muro, la variabile viene moltiplicata per −1, dirigendo la pallina verso sinistra. Questa parte emette anche un suono, per permettervi di udire la pallina che colpisce il muro.

La parte quattro usa la funzione ATTR per stabilire se la pallina ha demolito un mattone. Poichè i mattoni sono disegnati in giallo, viene segnato un punto ogni volta che la palla attraversa un carattere di colore giallo.

La quinta parte fa uso della funzione SCREEN$. La racchetta è disegnata con delle 'X' maiuscole, così quando la palla arriva sul fondo dello schermo, questa parte del programma controlla se la palla finisce su una 'X' e in caso contrario fa terminare il gioco.

Quando la palla colpisce la racchetta, deve essere fatta rimbalzare con un'angolazione casuale (mai comunque direttamente verso l'alto). Alcuni angoli sono semplici da calcolare, ma altri più ampi sono veramente complessi da elaborare, tanto da dare l'impressione che il gioco rallenti.

Ci sono diverse modifiche che potete fare al programma, se lo desiderate. Potreste ad esempio scrivere una routine che ogni tanto faccia rimbalzare la pallina sui mattoni rimasti, invece di distruggerli. Nel gioco non viene memorizzato il punteggio record, potrebbe essere un utile esercizio scrivere la parte del programma che esegue questo compito. Non ci sono caratteri definiti dall'utente. Se decidete di usare una racchetta disegnata da voi avrete dei problemi con la funzione SCREEN$; la stessa cosa succedeva in BOMBARDAMENTO, potete rifarvi a quel programma per superarli. Una pallina definita graficamente non dovrebbe invece essere molto complicata da realizzare. Potreste anche disegnare dei mattoni con una forma particolare, quella che volete, a patto però che li disegnate sempre in giallo (ricordate come lavora la funzione ATTR). I mattoni non devono avere necessariamente la stessa forma; potreste usare mattoni di forme diverse.

Ricordate che lo scopo del gioco è abbattere tutti i mattoni disegnati sullo schermo facendoci rimbalzare contro una pallina con la vostra racchetta. I tasti "5" e "8" muovono la racchetta rispettivamente verso sinistra e verso destra. Il gioco termina quando mancate la pallina. Per ogni mattone distrutto guadagnate un punto.

```
10 BORDER 0: PAPER 0: INK 0
20 CLS
25 INK 6
30 FOR a=1 TO 6
```

```
  40 FOR b=0 TO 31
  50 PRINT "■";
  60 NEXT b
  70 NEXT a
  80 LET u=21
  90 LET p=14
 140 INK 4
 150 LET a=u
 155 LET s=0
 160 LET t=p+1
 162 LET dt=(INT (RND*5)+1)/(INT
 (RND*5)+1)
 165 LET da=-1
 170 PRINT AT u,p; INVERSE 1;"XX
XXX"
 172 PRINT AT 11,0;s
 173 IF INKEY$="z" THEN COPY
 175 PRINT AT a,t;" "
 180 LET a=a+da
 190 LET t=t+dt
 195 IF a=0 OR a=21 OR t<=0 OR t
>=31 THEN GO TO 240
 197 IF ATTR (a,t)=6 THEN LET s=
s+1
 200 PRINT AT a,t; INVERSE 1; IN
K 1;"O"
 210 PRINT AT u,p;"     "
 220 LET p=p+((INKEY$="8")*(p<26
)-(INKEY$="5")*(p>0))*2
 225 LET p=p+(INKEY$="8")*(p=26)
-(INKEY$="5")*(p=25)
 230 GO TO 170
 240 IF a=21 AND SCREEN$ (a,t)="
X" THEN GO TO 162
 250 IF a=21 THEN GO TO 310
 260 IF t<=0 THEN LET t=ABS t: L
ET dt=-dt
 270 IF t>=31 THEN LET t=31: LET
 dt=-dt
280 BEEP .1,-10
 290 IF a=0 THEN LET da=1
 300 GO TO 200
 310 PRINT AT a,t; INK 2;"■"
```

```
320 BEEP 1,15
330 IF INKEY$<>"" THEN GO TO 33
0
340 IF INKEY$="" THEN GO TO 340
350 RUN
```

FEBBRE DA GIOCO

Questo programma, una variante della Slot Machine, fa veramente uso dei caratteri grafici definiti dall'utente (in gruppi di quattro) per creare effetti altamente spettacolari. Il listato del programma non riesce a dare l'idea di come appaia realmente questo programma su un televisore a colori.

Il concetto alla base del gioco è molto semplice: siete davanti ad una slot machine e disponete di 50$. La macchina è provvista di quattro finestre, dietro alle quali si trovano dei rulli rotanti. I rulli contengono tre simboli ripetuti disposti in ordine casuale: la MELA, la CILIEGIA e la CAMPANA. Quando premete ENTER i rulli cominciano a girare e quando si arrestano presentano ciascuno un simbolo, attraverso le finestrelle della macchina. Tre simboli uguali vi fanno vincere 35$ e quattro uguali 100$.

Il gioco continua sino a quando riuscite a sbancare la cassa (raggiungendo i 500$ o più) oppure quando terminate il vostro denaro. Ogni colpo vi costa 5$ che vengono automaticamente detratti dal vostro totale; potete vedere questo numero in alto a destra sullo schermo.

Di tanto in tanto entra in funzione il meccanismo HOLD (mantieni). La parola HOLD appare sullo schermo lampeggiante e indica che potete tenere fermo uno o più rulli nella prossima giocata. Per indicare quale rullo volete bloccare introducete un numero da 1 a 4 (i rulli sono numerati da sinistra verso destra) e poi premete ENTER. Potete bloccare quanti rulli volete (anche nessuno); per indicare che avete terminato la vostra scelta premete ENTER senza farlo precedere da alcun numero.

C'è un'altra caratteristica importante in questo gioco. La campana è il simbolo che ha maggior valore (ecco perchè vale la pena di conservarla quando capita l'opzione HOLD); se riuscite ad ottenere due campane affiancate ottenete un bonus di 15$ (questa regola non vale se escono quattro campane contemporaneamente). Se ad esempio ottenete tre campane, incassate 50$ invece dei normali 35. Se non ottenete tre simboli uguali ma riuscite ad avere due campane, una vicino all'altra guadagnate 15$. Il programma non evidenzia questo bonus, ma si limita ad aggiungere i 15$ al vostro totale.

Questo programma è progettato per avere una struttura che vi consenta di aggiungere qualunque modifica desideriate. Altri simboli potrebbero essere un'idea; se preferite potete variare i suoni del gioco o aggiungere un meccanismo NUDGE (cioè

la possibilità di dare un lieve colpetto ai rulli per migliorare la combinazione ottenuta); i possibili sviluppi di questo gioco sono limitati soltanto dalla vostra immaginazione.

Nel capitolo su come migliorare i vostri programmi, leggerete che spesso conviene definire i compiti principali del programma e poi scrivere delle routine che svolgano separatamente questi compiti; il programma principale provvederà poi a mandare in esecuzione tutte queste routine. Questo è proprio il metodo che ho seguito per scrivere questo programma.

Se osservate l'inizio del listato troverete tre chiamate ad altrettante routine, seguite da un GO TO 40 che riporta l'azione alla seconda chiamata. La prima delle tre routine (che inizia alla linea 720) assegna i valori iniziali alle variabili e definisce i caratteri grafici per i simboli. La seconda, che è chiamata durante tutto il programma, fa girare i rulli. Questa routine inizia alla linea 290 e chiama un'altra routine (che parte dalla linea 210) diverse volte nel corso di una mano (otto per l'esattezza). L'ultima routine, che viene mandata in esecuzione solo se il numero casuale generato alla linea 50 è maggiore di 45, attiva il meccanismo HOLD.

Potete vedere già dall'inizio del listato come è stato scritto questo programma. La prima cosa che feci fu decidere cosa mi serviva (una routine di inizializzazione, una routine di rotazione e una per l'opzione HOLD); poi scrissi le linee che chiamavano le varie routine. Alla fine il programma fu rinumerato, ma all'inizio avevo lasciato molto spazio tra una routine e l'altra; la routine di inizializzazione partiva dalla linea 9000, quella di rotazione dalla 8000, la routine di stampa dei rulli (che viene chiamata durante l'esecuzione della rotazione) iniziava alla linea 6000 e l'opzione HOLD alla linea 3000.

Noterete che il programma è suddiviso in sezioni da strisce di asterischi, inseriti in frasi REM. Questo sistema consente di stabilire facilmente quale sezione esegue un determinato compito e dovrebbe semplificarvi le cose se decidete di modificare il programma. Potete cambiare ad esempio la stampa dei rulli (lineee 210 - 280) senza modificare niente altro.

Lavorando in questo modo e applicando gli altri suggerimenti che troverete nel capitolo 'Guida a una migliore programmazione', scriverete dei programmi che richiederanno pochissime correzioni per essere messi in grado di funzionare e che saranno semplici da interpretare se tornerete ad esaminarli dopo un certo intervallo di tempo.

Esamineremo ora tutte le sezioni del programma, specificando per ognuna il compito svolto.

Abbiamo già parlato della prima sezione (linee 10-60), che rappresenta un ciclo di chiamate a diverse routine. Esamineremo queste routine nell'ordine in cui sono mandate in esecuzione invece di seguire l'ordine in cui sono scritte nel programma, perchè sarà più semplice comprendere quale compito sta portando a termine ciascuna di esse.

La prima routine inizia alla linea 720 e, come dice la frase REM che la precede, si occupa delle variabili e dei caratteri grafici. È bene assegnare sempre i valori iniziali

alle variabili in una sezione posta in fondo al programma; non solo per assicurarsi che il programma giri più velocemente, ma anche per lasciare quanto più spazio è possibile per aggiungerne di nuove, se la stesura del programma lo dovesse richiedere.

La variabile MONEY (denaro) contiene il totale dei soldi che avete a disposizione e viene inizializzata con un valore di 50. La linea 760 dimensiona quattro matrici, le prime due (che sono matrici di stringhe) contengono i caratteri grafici definiti e il loro colore (su questo punto torneremo in seguito), le altre due contengono i numeri generati per ogni rotazione (A) e quelli inseriti da voi per il meccanismo di HOLD (Q).

Le linee dalla 770 alla 1030 definiscono i caratteri grafici facendo uso dei dati contenuti nelle frasi DATA, a partire dalla linea 1050. Le linee 840 e 850 fondono assieme le quattro parti che formano la mela e i caratteri CHR$ 16 (carattere di controllo per attivare la INK) e CHR$ 2 (per ottenere il colore rosso). Potremo così far stampare la parte superiore della mela in rosso con l'istruzione PRINT A$ (1), invece di dover usare PRINT INK 2;"AB". Inoltre questo metodo rende più semplice richiamare il simbolo che a linea 440 seleziona con un numero casuale. Le linee 930 e 940 fondono le parti che compongono la campana e il colore giallo e la 1020 e la 1030 fanno lo stesso con la ciliegia (in realtà le ciliegie sono tre e non una) e il color magenta.

Le lettere che dovete inserire nel testo quando trovate "<A>" e "<B>" alla linea 840 sono naturalmente la A e la B grafiche, così per la "<C>" e la "<D>" alla linea 850, la "<E>" e la "<F>" alla 930, la "<G>" e la "<H>" alla 940, la "<J>" e la "<K>" alla 1020 e la "<L>" e la "<M>" alla 1030.

Le linee dalla 1040 alla 1130 stampano la slot machine. Il disegno dei rulli viene sovrastampato nel corso del gioco. (La forma della slot machine, il disegno dei simboli e la posizione dei messaggi che compaiono durante l'esecuzione del programma sono stati decisi con l'aiuto di un programma in vendita per lo Spectrum, Print'n'Plotter Jotter, che consente di agire sulle varie locazioni dello schermo con dei numeri di riferimento. Ho trovato questo prodotto di grande aiuto nel mio lavoro e penso che qualcosa del genere potrebbe risultare utile anche a voi).

Dopo l'esecuzione dell'istruzione RETURN, che restituisce il controllo al programma principale, viene richiamata la routine di rotazione dei rulli, che inizia alla linea 300. Prima di tutto vengono sottratti 5$ dal totale del vostro denaro (linea 320) per pagare la giocata. Successivamente la variabile PRIZE (che eventualmente conterrà la vostra vincita) viene posta a zero. La linea 350 vi chiede di premere ENTER per 'tirare la leva' della slot machine e la linea 360 attende che la tastiera sia rilasciata, poi il controllo passa alla linea 370 che resta in attesa di ENTER. La linea 380 cancella la scritta 'Premi ENTER...'. Il totale del vostro denaro (che ora è diminuito di 5$) viene stampato dalla linea 390; la parte di programma dalla linea 400 alla 470 fa ruotare i rulli 8 volte, richiamando la routine di stampa dei rulli (dalla linea 210 in avanti) ogni volta che viene eseguito un ciclo della variabile Z.

Il ciclo della variabile B, interno al ciclo Z, genera quattro numeri casuali compre-

si tra uno e quattro e li assegna agli elementi della matrice A. Se avete chiesto di bloccare qualcuno dei rulli (il programma può stabilirlo esaminando i valori della matrice Q) viene saltata la parte che genera il numero casuale e il controllo passa all'istruzione NEXT B. La linea 430 ha il compito di creare un simpatico rumore durante la rotazione. Notate che questo avviene dopo il controllo sulla matrice Q (che verifica se il rullo è bloccato) così si ottiene un suono differente se uno o più rulli risultano bloccati. Questo comunque risulterà più chiaro quando farete girare il programma.

La linea 480 pone a zero tutti gli ementi della matrice Q, il che equivale a disattivare il meccanismo di HOLD. Le linee dalla 490 alla 510 controlla se si è verificata una combinazione vincente; la linea 490 controlla il 'jackpot' (quattro simboli uguali); la linea 500 controlla se sono usciti tre simboli uguali e la 510 controlla se sono uscite due o più campane.

Le linee dalla 520 alla 540 producono diversi suoni mentre aspettate il risultato della giocata; la vostra eventuale vincita viene sommata alla variabile MONEY nella linea 550. La linea 570 stampa il tipo di vincita (Q$) e la linea 590 stampa il suo ammontare (PRIZE) sul pannello frontale della slot machine. Se vi è capitato un bonus (due o tre campane) le linee 610 e 620 provvedono a segnalarlo. La linea 630 produce un interessante effetto musicale per concludere la giocata. La linea 640 stampa il totale del denaro (MONEY) e le linee dalla 650 alla 670 cancellano i messaggi della vincita. Le linee 680 e 690 controllano se avete finito il denaro o se avete raggiunto lo scopo del gioco, sbancare la cassa (totalizzando più di 500$). Entrambe queste condizioni concludono il gioco.

La routine di stampa delle ruote (che parte della linea 210) è abbastanza logica; fa uso infatti degli elementi assegnati alla matrice A$ come descritto sopra quando si è parlato della grafica.

La routine di HOLD, l'ultima che prenderemo in esame, suona sette note (linea 100) poi stampa la parola HOLD! Questa parte del programma può apparire semplice, ma fate attenzione al controllo degli eventuali errori. Prima di tutto il computer attende l'ingresso di una stringa, invece di un numero anche se l'informazione di cui ha bisogno è di tipo numerico. Questo significa che premendo semplicemente ENTER il programma riprenderà l'esecuzione facendo accettare al computer un dato inesistente. La linea 130 controlla se l'input è stato nullo e se è così manda il controllo alla routine di cancellazione (seguendo il listato non avrete difficoltà a capire questo punto); viene quindi eseguita l'istruzione RETURN che conclude la routine.

Se l'input non era nullo la linea 140 prende il primo elemento della stringa – Q$(1) – e lo converte in un numero equivalente con l'istruzione VAL. Prendendo il primo elemento eludiamo la possibilità che l'utente introduca più di un numero alla volta. La linea 150 assegna questo numero all'elemento corrispondente della matrice Q (in altre parole, pone Q(1) uguale a uno o Q(4) uguale a quattro, e così via).

La linea 170 stampa, in una posizione relativa al numero introdotto 'Held>' e il numero poi rimanda l'azione alla linea 120 per l'input successivo. Quando il giocatore introduce una stringa nulla, premendo ENTER senza farlo precedere da un numero,

il computer va alla linea 190 che cancella i messaggi ed esegue un'istruzione RETURN per tornare al ciclo principale; qui la linea 60 rimanda l'esecuzione alla linea 40 e il ciclo riprende.

Vi suggerisco di inserire il programma così com'è, almeno all'inizio, e poi eventualmente lavorare sulle varie sezioni. Potreste desiderare di aggiungere altri simboli e altri tipi di vincite o, come ho suggerito all'inizio, un'opzione NUDGE. Una volta che avete capito come funziona la slot machine potete provare a scrivere un programma che usi quattro simboli e solo tre rulli, oppure uno che mostri anche i simboli sopra e sotto quelli usciti (avrete così nove simboli visualizzati nella versione con tre finestrelle) che consenta combinazioni vincenti anche in diagonale.

```
  10 REM FEBBRE DA GIOCO
  20 REM ****************
  30 GO SUB 720: REM VARIABILI
  40 GO SUB 290: REM ROTAZIONE
  50 IF RND>.45 THEN GO SUB 70 :
REM HOLD
  60 GO TO 40
  70 REM ************************
  80 REM       HOLD
  90 REM ************************
 100 FOR T=1 TO 7: BEEP .1,50-T*
7: NEXT T
 110 PRINT AT 9,22; FLASH 1; INK
 RND*4;"HOLD!"
 120 INPUT Q$
 130 IF Q$="" THEN GO TO 190
 140 LET Q=VAL Q$(1)
 150 LET Q(Q)=Q
 170 PRINT AT 9+Q,22; INK RND*4;
 FLASH 1;"Held> ";Q;" "
 180 GO TO 120
 190 PRINT AT 9,22;"     ";AT 10
,22;"        ";AT 11,22;"
  ";AT 12,22;"         ";AT 13,22
;"        "
 200 RETURN
 210 REM ********************
 220 REM STAMPA RUOTE
 230 REM ********************
 240 PRINT AT 6,3;A$(A(1));AT 7,
3;B$(A(1))
```

```
 250 PRINT AT 6,6;A$(A(2));AT 7,
6;B$(A(2))
 260 PRINT AT 6,9;A$(A(3));AT 7,
9;B$(A(3))
 270 PRINT AT 6,12;A$(A(4));AT 7
,12;B$(A(4))
 280 RETURN
 290 REM ********************
 300 REM     ROTAZIONE
 310 REM ********************
 320 LET MONEY=MONEY-5
 330 LET PRIZE=0
 340 LET Q$="": LET P$=""
 350 PRINT AT 4,19; FLASH 1; PAP
ER 1; INK 7;" Premi ENTER ";AT 5
,19;" per tirare la";AT 6,21;" l
eva"
 360 IF INKEY$<>"" THEN GO TO 36
0
 370 IF INKEY$="" THEN GO TO 370
 380 PRINT AT 4,19;"
 ";AT 5,19;"                ";AT 6,
21;"          "
 390 PRINT AT 2,28; FLASH 1; INV
ERSE 1; INK RND*4;MONEY; FLASH 0
; INVERSE 0;"  "
 400 FOR Z=1 TO 8
 410 FOR B=1 TO 4
 420 IF Q(B)=B THEN GO TO 450
 430 BEEP .008,Z*B
 440 LET A(B)=INT (RND*3)+1
 450 NEXT B
 460 GO SUB 210
 470 NEXT Z
 480 DIM Q(4)
 490 IF A(1)=A(2) AND A(2)=A(3)
AND A(3)=A(4) THEN LET Q$=" JACK
POT!! ": LET PRIZE=100: GO TO 52
0
 500 IF (A(1)=A(3) AND A(3)=A(4)
) OR (A(1)=A(2) AND A(2)=A(3)) O
R (A(2)=A(3) AND A(3)=A(4)) OR (
A(2)=A(3) AND A(3)=A(4)) OR (A(1
```

```
)=A(2) AND A(2)=A(4))  THEN LET
Q$="3 DI UN TIPO!": LET PRIZE=35
 510 IF A(1)=2 AND A(2)=2 OR A(2
)=2 AND A(3)=2 OR A(3)=2 AND A(4
)=2 THEN LET P$=" BONUS!! ": LET
 PRIZE=PRIZE+15
 520 FOR G=1 TO 30
 530 BEEP .008,50-G: BEEP .008,5
0
 540 NEXT G
 550 LET MONEY=MONEY+PRIZE
 560 IF Q$="" THEN GO TO 630
 570 PRINT AT 19,6; BRIGHT 1; FL
ASH 1; INK 2; PAPER 4;Q$
 580 PRINT TAB 6; BRIGHT 1; FLAS
H 1; INVERSE 1; INK 2;  PAPER 4;
Q$
 590 IF PRIZE>0 THEN PRINT AT 10
,3; BRIGHT 1; INK 2;"VINCI $";PR
IZE
 600 IF P$="" THEN GO TO 630
 610 PRINT AT 10,20; FLASH 1; IN
K 5; PAPER 3;P$
 620 PRINT AT 11,20; INVERSE 1;
FLASH 1; INK 3;P$
 630 FOR T=1 TO 100: BEEP .008,T
-40: NEXT T
 640 PRINT AT 2,28; FLASH 1; INV
ERSE 1; INK RND*4;MONEY; INVERSE
 0
 650 PRINT AT 19,6;"
   ";TAB 6;"                  "
 660 PRINT AT 10,2; INK 0;"
"
 670 PRINT AT 10,20;"          ";
AT 11,20;"          "
 680 IF MONEY<1 THEN PRINT AT 0,
0; FLASH 1;"E' LA FINE, AMICO MI
O","SEI FALLITO": STOP
 690 IF MONEY>499 THEN PRINT AT
0,0; FLASH 1;"MI HAI SBANCATO!!"
: STOP
 700 RETURN
```

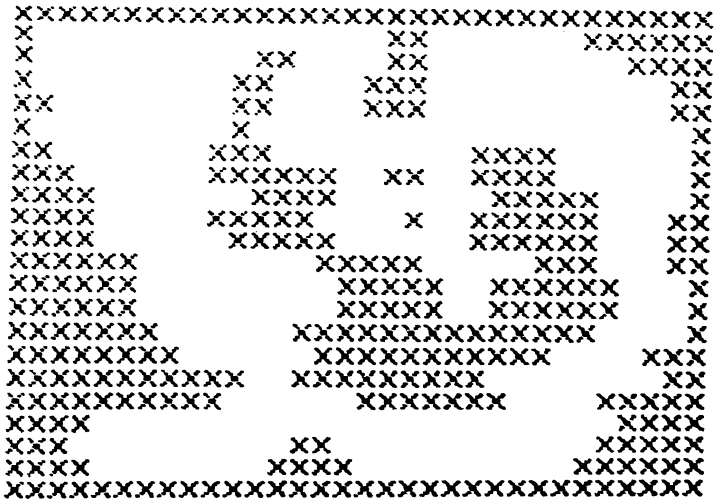
```
 720 REM ********************
 730 REM  VARIABILI/GRAFICA
 740 REM ********************
 750 LET MONEY=50
 760 DIM A$(3,4): DIM B$(3,4): D
IM A(4): DIM Q(4)
 770 FOR Z=0 TO 7
 780 READ A: READ B: READ C: REA
D D
 790 POKE USR "A"+Z,A
 800 POKE USR "B"+Z,B
 810 POKE USR "C"+Z,C
 820 POKE USR "D"+Z,D
 830 NEXT Z
 840 LET A$(1)=CHR$ 16+CHR$ 2+"<
A>"+"<B>"
 850 LET B$(1)=CHR$ 16+CHR$ 2+"<
C>"+"<D>"
 860 FOR Z=0 TO 7
 870 READ E: READ F: READ G: REA
D H
 880 POKE USR "E"+Z,E
 890 POKE USR "F"+Z,F
 900 POKE USR "G"+Z,G
 910 POKE USR "H"+Z,H
 920 NEXT Z
 930 LET A$(2)=CHR$ 16+CHR$ 6+"<
E>"+"<F>"
 940 LET B$(2)=CHR$ 16+CHR$ 6+"<
G>"+"<H>"
 950 FOR Z=0 TO 7
 960 READ J: READ K: READ L: REA
D M
 970 POKE USR "J"+Z,J
 980 POKE USR "K"+Z,K
 990 POKE USR "L"+Z,L
1000 POKE USR "M"+Z,M
1010 NEXT Z
1020 LET A$(3)=CHR$ 16+CHR$ 3+"<
J>"+"<K>"
1030 LET B$(3)=CHR$ 16+CHR$ 3+"<
L>"+"<M>"
1040 PAPER 7: CLS : BORDER 7
```

```
1050 PRINT AT 1,3; INK 0;"[redacted]
[redacted]"
1060 PRINT TAB 2; INK 0;"[redacted]"; IN
K 4; PAPER 2; FLASH 1;"[redacted]
"; FLASH 0; PAPER 7; INK 0;"[redacted]
[redacted]";TAB 21; INK 2; FLASH 1;"CONTA
NTI $";MONEY
1070 PRINT TAB 2; INK 0;"[redacted]"; IN
VERSE 1; INK 4; PAPER 2; FLASH 1
;"[redacted]"; INVERSE 0; FLASH 0
; PAPER 7; INK 0;"[redacted]  [redacted]"
1080 FOR Z=1 TO 11
1090 PRINT TAB 2;"[redacted]
[redacted]"
1100 NEXT Z
1110 PRINT TAB 2;"[redacted]"; PAPER 6; I
NK 3; FLASH 1;"[redacted]"; FLA
SH 0; PAPER 7; INK 0;"[redacted]"
1120 PRINT TAB 2;"[redacted]"; PAPER 6; I
NK 3; INVERSE 1; FLASH 1;"[redacted]
[redacted]"; FLASH 0; INVERSE 0; PAPE
R 7; INK 0;"[redacted]"
1130 PRINT TAB 2;"[redacted]
[redacted]"
1140 RETURN
1150 REM MELA
1160 DATA 0,8,121,254,0,28,63,25
4,0,48,63,254,15,32,31,254
1170 DATA 31,240,15,252,51,252,7
,248,52,254,1,224,101,254,0,0
1180 REM CAMPANA
1190 DATA 0,128,7,240,0,128,15,2
48,1,192,15,248,3,224,31,252
1200 DATA 3,224,0,128,3,224,1,19
2,3,224,1,192,3,224,0,0
1210 REM CILIEGIA
1220 DATA 0,0,1,136,0,2,26,8,56,
12,60,16,124,52,126,208
1230 DATA 127,204,61,224,124,20,
25,240,124,36,1,224,56,72,0,224
1240 STOP
```

CORSA SCREEN$

Ho chiamato questo programma CORSA SCREEN$ perchè la funzione SCREEN$ dello Spectrum è la chiave del programma.

L'idea di base è semplice. Sullo schermo compare una pista; partite nell'angolo in alto a sinistra dello schermo, la vostra posizione è segnalata da un 9 lampeggiante. Dovete pilotare questo numero lampeggiante lungo il tracciato in direzione oraria, usando i tasti 'R' (su), 'F' (sinistra), 'J' (destra) e 'N' (giù). Scoprirete presto che è molto facile pilotare con questi tasti; essi sono stati scelti perchè le vostre dita vengano a trovarsi in un una posizione più naturale (in effetti sono più comodi dei tasti '5', '6', '7' e '8', usati generalmente per controllare i movimenti sullo schermo).



Provate a muovervi lungo il tracciato, senza colpire i muri. Quando sarete tornati nell'angolo in alto a sinistra, il nove diventerà un otto e in seguito continuerà a decrementare ogni volta che completate un circuito. Il vostro scopo è completare il tracciato senza incidenti per nove volte.

Quando fate un incidente vi viene dato un punteggio che è in relazione al tempo in cui siete riusciti a pilotare e al vostro punteggio totale. Il punteggio viene modificato in base al tempo in cui siete rimasto senza incidenti in quel particolare giro. Il punteggio viene decrementato per tutto il tempo della gara, così è nel vostro interesse guidare velocemente e non essere troppo prudenti. Prima di fare girare il programma assicuratevi che il CAPS LOCK sia inserito, così la funzione INKEY$ interpreterà gli input correttamente.

```
  10 REM CORSA SCREEN$
  20 REM INSERIRE CAPS LOCK
  30 GO SUB 360
  40 FOR Z=9 TO 1 STEP -1
  50 PRINT AT CARD,CARA; BRIGHT
1; FLASH 1;Z
  60 LET SCORE=SCORE-1
  70 LET A1=CARA: LET D1=CARD
  80 IF INKEY$="F" AND CARA>1 TH
EN LET CARA=CARA-1
  90 IF INKEY$="J" AND CARA<30 T
HEN LET CARA=CARA+1
 100 IF INKEY$="R" AND CARD>1 TH
EN LET CARD=CARD-1
 110 IF INKEY$="N" AND CARD<20 T
HEN LET CARD=CARD+1
 120 PRINT AT D1,A1;" "
 130 IF SCREEN$ (CARD,CARA)="X"
THEN GO TO 170
 140 IF CARA<4 AND CARD<4 THEN L
ET CARA=4: LET CARD=1: PRINT AT
D1,A1;" ": BEEP .4,RND*50: NEXT
Z
 150 IF Z=0 THEN GO TO 270
 160 GO TO 50
 170 REM *****CRASH*****
 180 FOR R=1 TO 50: BORDER RND*7
 190 PRINT AT CARD,CARA-1; BRIGH
T 1;"X*X";AT CARD,CARA-1; INVERS
E 1;"*X*"; INVERSE 0
 200 NEXT R: BORDER 2
 210 PRINT AT 3,5; FLASH 1;"Punt
i: "; INVERSE 1;INT (SCORE*3-127
*Z)
 220 FOR R=1 TO 50: BEEP .008,R/
50: BEEP .007,50/R: NEXT R
 240 CLS
 250 GO SUB 400
 260 GO TO 40
 270 REM FINE GARA
 280 PRINT AT 5,0; FLASH 1; BRIG
HT 1;"Ce l'hai fatta campione!!"
 290 PRINT INVERSE 1; FLASH 1; B
```

```
RIGHT 1;''TAB 5;"ATTENDI"
 300 FOR G=1 TO 300: NEXT G
 310 CLS
 320 LET SCORE=10000
 330 GO SUB 400
 340 GO TO 40
 350 STOP
 360 REM INIZIALIZZAZIONE
 370 PAPER 0: INK 7: CLS
 380 BORDER 2
 390 LET SCORE=5000
 400 PRINT "Punteggio di partenz
a "; FLASH 1;SCORE: PAUSE 300
 420 FOR Y=1 TO 0 STEP -1
 430 INVERSE Y
 440 PRINT AT 0,0;"XXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXX"
 450 PRINT "X                  XX
      XXXXXX"
 460 PRINT "X           XX     XX
        XXXX"
 470 PRINT "X          XX     XXX
         XX"
 480 PRINT "XX         XX     XXX
         XX"
 490 PRINT "X          X
         X"
 500 PRINT "XX        XXX
 XXXX       X"
 510 PRINT "XXX       XXXXXX  XX
 XXXX       X"
 520 PRINT "XXXX        XXXX
 XXXXX      X"
 530 PRINT "XXXX      XXXXX    X
 XXXXXX   XX"
 540 PRINT "XXXX       XXXXX
 XXXXXX   XX"
 550 PRINT "XXXXXX         XXXXX
    XXX   XX"
 560 PRINT "XXXXXX           XXXXX
  XXXXXX   X"
 570 PRINT "XXXXXX           XXXXX
  XXXXXX   X"
```

```
 580 PRINT "XXXXXXX       XXXXXXX
XXXXXXX     X"
 590 PRINT "XXXXXXXX       XXXXXX
XXXXX     XXX"
 600 PRINT "XXXXXXXXXXX  XXXXXXX
XX          XX"
 610 PRINT "XXXXXXXXXX       XXXX
XXX     XXXXX"
 620 PRINT "XXXX
          XXXX"
 630 PRINT "XXX            XX
        XXXXX"
 640 PRINT "XXXX          XXXX
      XXXXXX"
 650 PRINT "XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXX"
 660 NEXT Y
 670 LET CARA=4
 680 LET CARD=1
 690 RETURN
 700 REM  R = SU
 710 REM  F = SINISTRA
 720 REM  J = DESTRA
 730 REM  N = GIU'
 740 STOP
```

La prima volta che fate girare il programma viene mandata in esecuzione la routine che inizia alla linea 369. Questa routine, come molte di quelle che si trovano in fondo ai programmi di questo libro, inizializza le variabili necessarie. Il colore dello sfondo è il nero, il testo è in bianco e il bordo dello schermo viene colorato in rosso. Il punteggio iniziale è posto a 5000. La linea 400 stampa il punteggio iniziale sullo schermo e lo mantiene visibile per qualche secondo.

Il ciclo della variabile Y (linee 420-660) stampa il tracciato due volte, il che dà maggiore risalto alla partenza come vedrete quando farete girare il programma. La coordinata orizzontale della macchina sullo schermo è memorizzata nella variabile CARA, mentre CARD rappresenta quella verticale.

Quando il controllo torna dalla routine al programma principale, inizia un ciclo (Z) che conta all'indietro da nove a uno. Viene stampata la macchina (linea 50) e il punteggio viene decrementato di uno (linea 60). La linea 70 memorizza le coordinate della macchina in due nuove variabili (A1 e D1), così la macchina può essere cancellata una volta che si sia spostata. Le linee da 80 a 110 leggono la tastiera, la linea

120 cancella la macchina dalla sua vecchia posizione. La linea 140 controlla se la macchina è tornata nell'angolo in alto a sinistra del tracciato e se è così rimanda la azione al prossimo giro, tramite il ciclo Z. La linea 150 controlla se il ciclo Z è finito (cioè se avete vinto), e se questo è accaduto il controllo passa alla routine FINE GARA, che inizia alla linea 270. Se Z non è uguale a zero (cioè non è ancora stato completato l'ultimo percorso), la linea 160 rimanda il controllo alla linea 50, e si prosegue nello stesso ciclo Z; la macchina è stampata nella sua nuova posizione e la gara continua.

Non abbiamo parlato della linea 130, perchè volevo dedicarle un po' più di tempo rispetto alle altre linee e anche perchè uno degli effetti della linea 130 è quello di rimandare l'azione alla parte di programma che segue l'ultimo punto di cui abbiamo parlato, la routine di CRASH che inizia alla linea 170. Diamo un'occhiata alla linea 130 essa fa uso della funzione SCREEN$ per eseguire uno dei due compiti che questa istruzione svolge sullo Spectrum, cioè esaminare il contenuto di una specifica locazione dello schermo (l'altra funzione è quella di consentire la memorizzazione su nastro di una videata completa). In alcuni dei giochi precedenti (come CAMPO DI TESCHI e JACKMAN), la funzione è stata usata per entrambi gli scopi.

I due numeri che seguono la parola SCREEN$ compresi tra parentesi, sono le coordinate della locazione dello schermo che vogliamo esaminare e sono le stesse che useremmo con l'istruzione PRINT AT. La linea 130 esamina la posizione CARD, CARA che è quella dove dovrebbe essere stampato il prossimo numero lampeggiante, che rappresenta la macchina. Se in questa posizione viene trovata una 'X', il programma capisce che l'auto è andata a sbattere, così l'azione prosegue con la routine CRASH che inizia alla linea 170. Se non viene trovata una 'X' il programma prosegue nel ciclo. Se decidete di modificare il tracciato della pista usando un carattere diverso da 'X' assicuratevi di modificarlo anche alla linea 130 (e ricordate che la funzione SCREEN$ non legge i caratteri grafici).

La routine CRASH è interessante, perchè produce alcuni effetti pirotecnici che potrete impiegare anche in altri programmi. Il ciclo R (linee 180 - 200) stampa un disegno per dare l'impressione dell'incidente ('X' alternato con '*') mentre il colore del bordo cambia in modo casuale.

Questo cambio del colore del bordo è piuttosto allarmante e potrebbe anche, come accade nel mio televisore, causare delle modifiche al disegno centrale oppure produrre qualche ronzio dall'apparecchio televisivo. Terminato il ciclo R, il colore del bordo torna rosso (seconda istruzione della linea 200), poi il punteggio è stampato sullo schermo, calcolandolo con una particolare formula (il punteggio moltiplicato tre volte, meno 127 per Z). La linea 220 produce un ciclo per emettere dei suoni, qualcosa del tipo discusso nel capitolo dei suoni di questo libro. La linea 250 rimanda l'azione alla linea seguente a quella in cui il punteggio veniva inizializzato, così il vostro nuovo punteggio non viene modificato.

La routine FINE GARA, che parte alla linea 270, viene eseguita solo se riuscite a compiere tutti e nove i giri, senza incidenti, un risultato veramente formidabile. Può essere fatto, ma richiede molta pratica. "Ce l'hai fatta campione!!" è il messaggio

che compare sullo schermo e il vostro punteggio viene portato a 10000. Da qui il programma prosegue nello stesso modo in cui termina la routine CRASH.

Ci sono molte cose che potreste cambiare in questo programma. Potreste voler definire un carattere grafico per rappresentare la macchina e stabilire un solo giro come obiettivo del gioco. Potreste anche aggiungere qualche altro effetto sonoro.

Non sentitevi costretti a seguire il mio tracciato. Prima provatelo e poi se volete modificatelo o complicatelo in qualunque modo vi sembri opportuno. Potreste persino scrivere una routine che generi un nuovo tracciato ad ogni giro, casualmente, oppure che alterni due o più tracciati. Scoprirete che è abbastanza facile barare in questo programma; basta semplicemente guidare l'auto indietro verso l'angolo in alto a sinistra e tenercela mentre viene decrementato il ciclo Z. Potreste scrivere una routine per evitare questo tipo di imbroglio.

COLONIA

Colonia è una variante del famoso gioco LIFE di John Conway. In questa versione un certo numero di piccole rane verdi si muovono sullo schermo, ricreando il modello con cui LIFE divenne famoso. Il gioco LIFE fu inventato da mister Conway della Canbridge University, nell'ottobre del 1970. Esso simulava la nascita, la crescita e la morte delle cellule in una colonia chiusa. Questo comunque non è un incubo maltusiano, in cui le cellule si riproducono incessantemente, fino a uscire dallo schermo. Nel mondo di LIFE e nel suo derivato COLONIA, nascita, crescita e morte sono governate da regole più civili.

Le cellule vivono in una grigilia e seguono queste regole stabilite da Conway:

- Ci sono al massimo otto cellule circostanti per ogni cellula della griglia
- La sopravvivenza della prossima generazione richiede che una cellula abbia due o tre cellule circostanti, né di più né di meno.
- Se ci sono tre cellule circostanti in una posizione vuota della griglia, nascerà una nuova cellula nella prossima generazione.
- Ogni cellula con quattro o più cellule circostanti morirà nella prossima generazione.

Ci sono molti modi per scrivere questo programma, ma vi suggerisco di provare a scriverlo voi stessi prima di vedere come ho fatto io per la mia colonia di rane. Comunque avete bisogno di un'altra informazione per costruire il gioco in modo appropriato: le regole devono essere applicate in tutta la griglia nello stesso momento, così i cambiamenti che influiranno sulla prossima generazione non influenzeranno le cellule della generazione attuale che non sono ancora state esaminate. Costruite una griglia di dieci per dieci e provate a scrivere un programma che disponga le cel-

lule su di essa e le controlli in accordo con le regole di Conway; per fare questo avrete bisogno di una routine che vi consenta di 'circumnavigare' la griglia per controllare le varie posizioni.

Questo che segue è il listato del mio programma, che potreste voler usare come punto di partenza per il vostro; se preferite potete solo farlo girare per vedere come agisce in pratica, rendendovi conto di qual è l'obiettivo del gioco.

```
  10 REM ********************
  20 REM     COLONIA
  30 REM ********************
  40 GO SUB 260
  50 BEEP .008,RND*40: BORDER RN
D*7
  60 PRINT AT 2,6; INK 2; FLASH
1; BRIGHT 1;"GENERAZIONE";Z;" ";
AT 5,0;
  70 FOR M=2 TO 9: PRINT 'TAB 6;
  80 FOR N=2 TO 9
  90 LET X(M;N)=Y(M,N)
 100 PRINT INK 4;CHR$ (X(M,N));"
 ";
 110 NEXT N
 120 PRINT
 130 NEXT M
 140 FOR M=2 TO 9
 150 FOR N=2 TO 9
 160 LET Q=0
 170 FOR W=1 TO 8
 180 LET Q=Q+(VAL A$(W)=144)
 190 NEXT W
 200 IF X(M,N)=144 AND Q<>2 AND
Q<>3 THEN LET Y(M,N)=32
 210 IF X(M,N)=32 AND Q=3 THEN L
ET Y(M,N)=144
 220 NEXT N
 230 NEXT M
 240 LET Z=Z+1
 250 GO TO 50
 260 REM ** INIZIALIZZAZIONE **
 270 RESTORE
 280 DIM A$(8,10): DIM X(10,10):
```

```
 DIM Y(10,10)
 290 FOR Q=1 TO 8
 300 READ Q$
 310 LET A$(Q)=Q$
 320 NEXT Q
 330 DATA "X(M-1,N-1)","X(M-1,N)
","X(M-1,N+1)","X(M,N-1)","X(M,N
+1)","X(M+1,N-1)","X(M+1,N)","X(
M+1,N+1)"
 340 REM COLTURA INIZIALE
 350 FOR P=1 TO 10
 360 FOR Q=1 TO 10
 370 LET X(P,Q)=32: LET Y(P,Q)=3
2
 380 NEXT Q
 390 NEXT P
 400 FOR Q=1 TO 15
 410 READ A
 420 READ B
 430 LET X(A,B)=144: LET Y(A,B)=
144
 440 NEXT Q
 450 DATA 3,4,3,5,3,6,4,3,4,5,4,
7,5,4,5,5,5,6,6,3,6,5,6,7,7,4,7,
5,7,6
 460 FOR Q=0 TO 7
 470 READ P
 480 POKE USR "A"+Q,P
 490 NEXT Q
 500 DATA 66,153,90,60,24,126,66
,102
 510 LET Z=1
 520 RETURN
```

Esaminerò tutto il programma per spiegarvi cosa fa ogni sezione di esso. Andremo prima alla routine della linea 260, dove vengono eseguite le necessarie inizializzazioni. La linea 280 dimensiona tre matrici: A$ per contenere le relazioni matematiche che intercorrono fra una posizione della griglia e le posizioni circostanti; X per contenere la generazione attuale e Y per contenere la prossima generazione che viene formata mentre la generazione attuale si trova sullo schermo.

Il ciclo dalla linea 290 alla 320 legge i dati della linea 330 e li memorizza negli elementi di Q$. Le linee da 350 a 390 riempiono la griglia con spazi (32 è il codice dello Spectrum per uno spazio vuoto), lo stesso viene fatto per la 'prossima generazione'. Le linee da 400 a 440 leggono le informazioni relative alla prima generazione contenute nella frase DATA alla linea 450 e li memorizzano in entrambe le griglie. Il numero che esse assegnano agli elementi delle matrici è 144, il codice del primo carattere grafico definito (A grafica).

Il ciclo finale Q, linee 460 - 490, definisce la forma della rana che sarà stampata come CHR$ 144 al momento opportuno. Z, il numero della generazione, è inizializzato a uno.

Al ritorno della routine di inizializzazione c'è un BEEP e il bordo lampeggia. Il numero della generazione è stampato in alto sullo schermo e il ciclo dalla linea 70 alla 130 stampa la colonia. La linea 90 legge le informazioni per la prossima generazione (matrice Y) dalla generazione attuale (matrice X). Poi la matrice X viene stampata.

Il prossimo ciclo, linee 140 - 230, è il più importante del programma. Q, posto a zero alla linea 160, è la variabile usata per contare le cellule circostanti a quella presa in considerazione. Il ciclo W, linee 170-190, è il ciclo magico che controlla le otto cellule circostanti alla cellula presa in considerazione. Notate che M e N, i due contatori del ciclo di controllo, variano solo da due a nove, così le cellule che si trovano nelle posizioni più esterne (e che non sono quindi da considerarsi circostanti) non vengono controllate. La linea 180 incrementa il valore di Q ogni volta che l'espressione VAL A$(W)=144 risulta vera. Lo Spectrum assegna il valore uno a un'espressione che risulti vera, così questa linea esegue lo stesso compito di:

```
IF VAL A$(W)=144 THEN LET Q=Q+1
```

in modo più breve.

Una volta che le cellule circostanti sono state controllate, il programma prosegue con il conteggio. Se la cellula in considerazione contiene una rana (contiene cioè il valore 144) e non ci sono due o tre componenti (Q non è uguale a due o tre) la cellula corrispondente della prossima generazione è posta a 32 (matrice Y). Se comunque la cellula controllata è vuota (contiene cioè il valore 32) e ci sono tre componenti (Q uguale a tre) l'elemento della prossima generazione è posto a 144 e avviene una nascita.

Una volta che questo processo si è ripetuto per l'intera griglia (eccettuate le cellule dei lati esterni), il conteggio della generazione è incrementato di uno (linea 240) e il programma torna alla linea 50 per stampare la nuova generazione e il ciclo riprende.

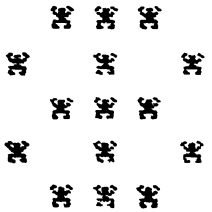
Potete aggiungere la vostra colonia iniziale modificando la frase DATA alla linea

450 dove ogni coppia di numeri rappresenta la posizione di una rana (vedi linea 430). Un modello che potete provare dopo aver fatto girare il programma originale è:

```
450 DATA 3,3,3,5,3,6,3,8,6,3,6,
8,9,5,9,6,4,4,4,7,5,4,5,7,6,5,6,
6,7,5,7,6,8,3,8,4,8,7,8,8,9,3,9,
4,9,7,9,8
```

notate che ci sono ventiquattro elementi in questa generazione, così dovete cambiare il 15 alla fine della linea 400 in 24. Questo modello prosegue per sedici generazioni prima di morire definitivamente. Una volta che avete visto le figure particolarmente attraenti che si formano in queste sedici generazioni, aggiungete una singola cellula alla colonia originale, oppure cancellatene una e osservatene gli effetti singolari che si ripercuotono su ogni generazione.

Provate a scrivere una routine che stabilisca casualmente la colonia iniziale oppure un'altra che vi consenta di introdurla da tastiera, all'inizio di ogni esecuzione. Quando elaborate la colonia inziale, formate una griglia di dieci per dieci e numerate le posizioni da uno a dieci dall'alto verso il basso e da sinistra a destra e caricateci dentro i vostri numeri. Non inserite nessuna cellula di partenza in una posizione esterna della griglia.



LAVORANDO PER IL CAPO — una simulazione

In LAVORANDO PER IL CAPO siete il direttore di una fabbrica che produce lampadine. Avete il controllo quasi completo sul personale, sulla produttività e sui prezzi di vendita.

Il vostro compito è restare in affari il più a lungo possibile e possibilmente raggiungere un milione di dollari.

Questo programma è una simulazione in cui il computer assume il ruolo di 'ambiente reale'; manipola le informazioni esattamente come accade nella realtà, anche se con qualche limitazione.

Le due parole chiave per creare una simulazione sono 'replicare' e 'semplificare'. Tentiamo di replicare la vita reale, ma siccome non c'è modo di includere tutti i fattori considerabili, dobbiamo introdurre alcune semplificazioni per ottenere un gruppo di variabili con cui lavorare.

In questo programma, che vi fornisce tutte le informazioni necessarie durante l'esecuzione, vi trovate a capo di una fabbrica di lampadine. All'inizio del gioco dovete dichiarare quante persone lavorano per voi, quanto sono pagate ogni settimana, di quanto capitale disponete, che scorta di magazzino avete e qual è il prezzo di vendita delle lampadine. Il programma vi restituisce il conto totale delle spese e questa è una cifra che dovete sempre tenere a mente. Il programma lavora per settimane e il vostro compito è riuscire a pagare tutte le spese e gli ordini di materie prime ogni settimana, o andrete in fallimento.

La prima scelta che dovete fare è se assumere o licenziare personale. Mentre il sindacato dei vostri lavoratori è molto soddisfatto se assumete personale, possono esserci resistenze se decidete di licenziare; così malgrado le vostre intenzioni, potete liberarvi soltanto del numero di impiegati che vi consente il sindacato. Se introducete '7' in risposta a quanto personale licenziare, il computer potrebbe rispondere 'il sindacato ti consente di licenziare 2 persone'; questo significa, come nella vita reale, che non avete il controllo completo sulle dimensioni del vostro personale. Dovete inoltre tenere presente che una persona può produrre un numero limitato di lampadine ogni settimana, così la vostra fabbrica non sopravviverà se riducete il personale ad un solo operaio.

Di tanto in tanto i lavoratori chiedono un aumento di stipendio e voi non avete altra scelta che pagare. Inoltre i vostri fornitori sono noti per applicare aumenti di prezzo sfrenati sulle materie prime, quindi conducete una battaglia continua contro l'aumento dei prezzi e delle spese.

Ci sono tre fattori che potete controllare, anche se con qualche limitazione:

1) Il numero di persone che lavorano per voi (non potete licenziare a vostro piacimento; d'altra parte ogni membro del personale vi costa denaro ogni settimana e la produzione massima per persona ha un limite ben definito).
2) Il numero di lampadine che producete ogni settimana (stabilite un obiettivo da raggiungereg ma raramente esso viene raggiunto).

3) Il prezzo di vendita può essere modificato secondo i vostri desideri (ma ogni aumento rende più difficile la vendita della scorta che avete in magazzino; non è detto che riusciate a vendere ogni settimana tutta la produzione di lampadine e il quantitativo venduto è in parte legato alle modifiche che apportate ai prezzi).

Dato che dovete pagare i vostri conti, di tanto in tanto dovete alzare i prezzi (perchè le spese aumentano e lo stesso accade con i costi delle materie prime) ma la consapevolezza che ogni cambiamento di prezzo diminuisce il potere di vendita dei vostri prodotti, dovrebbe moderarvi nei vostri aumenti.

Può darsi che questa descrizione vi abbia lasciati un po' perplessi, ma non preoccupatevi. Non dovete ricordare tutto ciò che è stato dettog il programma esegue la maggior parte del lavoro e vi darà tutte le informazioni che vi servono.

LAVORANDO PER IL CAPO può finire in due modi. Il più comune è la bancarotta, nel quale caso vi verrà detto il numero di settimane per il quale siete riusciti a condurre la vostra fabbrica. Il secondo modo è che accumuliate un milione di dollari (capitale più merce in magazzino). Come scoprirete durante il gioco tutto è contro di voi. C'è un detto in affari: il primo compito di un'operazione finanziaria è quello di sopravvivere. Scoprirete quanto questo sia vero quando farete girare il programma. Malgrado le grandi semplificazioni che si sono rese necessarie per ottenere una simulazione sufficientemente agile, essa replica la realtà in maniera impressionante. Vi scoprirete a disperarvi quando leggete le cifre delle vendite e della produzione, agonizzerete per prendere decisioni sul livello dei prezzi, sui licenziamenti e sulle assunzioni.

Malgrado la sua esasperante lentezza, LAVORANDO PER IL CAPO è veramente un programma semplice e una volta che avete afferrato il suo svolgimento troverete relativamente facile creare altre simulazioni di qualunque tipo per vostro conto.

Come suggerito diverse volte nel libro, potete creare dei programmi molto lunghi dividendoli in piccole routine, richiamate da un programma principale e rendendo così il tutto più semplice e veloce. Questo è il sistema con cui è stato scritto LAVORANDO PER IL CAPO. Segue prima di tutto un listato del programma, poi troverete una discussione su di esso.

```
 10 REM ************************
 20 REM LAVORANDO PER IL CAPO
 30 REM ************************
 40 GO SUB 9000: REM VARIABILI
 50 LET WEEK=WEEK+1
100 GO SUB 5000: REM STAMPA
110 GO SUB 6000: REM PERSONALE
120 GO SUB 5000: REM STAMPA
130 GO SUB 5130: REM PRODUZ.
140 GO SUB 5000: REM STAMPA
```

```
 150 GO SUB 4000: REM VENDITA
 160 GO SUB 3000: REM IMPREVEDIB
ILI
 170 LET CAPITAL=CAPITAL-WAGE*WO
RKFORCE
 180 GO TO 50
2990 REM **********************
3000 REM IMPREVEDIBILE
3005 CLS
3010 IF RND<.45 THEN GO TO 3100
3020 LET A=INT (RND*7)+1
3025 BEEP .5,RND*50: BEEP .4,RND
*50
3030 PRINT ''"I sindacati chiedo
no un aumento del ";A;"%"
3040 LET WAGE=INT (100*(WAGE+(A*
WAGE/100)))/100
3050 PAUSE 100
3060 PRINT ''''"La paga singola a
ttuale e' $";WAGE
3070 PAUSE 100
3075 BEEP .5,RND*50: BEEP .4,RND
*50
3080 CLS
3100 IF RND<.85 THEN GO TO 3190
3110 PRINT ''' INK 2; FLASH 1;"U
na inondazione","ha danneggiato
parte dello stock",,,,"Attendere
 il rapporto sui danni",
3120 PAUSE 100
3130 LET A=INT (RND*STOCK/2)+1
3140 LET STOCK=STOCK-A
3150 PRINT '' INK 1; FLASH 1;"So
no rimaste distrutte ",A;"lampad
ine","per un valore di $";A*SELL
PRICE
3160 PAUSE 100
3170 PRINT ''"Lo stock in magazz
ino ammonta a";STOCK;" lampadine
"
3180 PAUSE 100
3190 IF RND>.3 THEN GO TO 3290
3195 CLS
```

```
3200 PRINT ''"I fornitori annunc
iano forti    aumenti di prezzo!
"
3210 PAUSE 100
3220 LET A=INT ((RND*100*COST/7)
)/100
3225 IF A<.01 THEN GO TO 3220
3230 PRINT ''"Il costo di produz
ione per ogni","lampadina e' sal
ito di "; FLASH 1; INK 2;"$";A;
FLASH 0
3240 PAUSE 100
3250 LET COST=COST+A
3260 PRINT '' INK 7; PAPER 1;"Ad
esso il costo ammonta a "; FLASH
 1; INK 7; PAPER 0;"$";COST;" ";
3270 PRINT "per ciascuna"
3280 PAUSE 100
3290 IF RND<.65 AND MAKE<SELLPRI
CE THEN RETURN
3300 CLS
3305 BEEP .5,RND*50: BEEP .4,RND
*50
3310 PRINT ''"Hai l'opportunita'
 di alzare","il prezzo"
3320 PRINT '"Attualmente si vend
ono a $ ";SELLPRICE
3330 PAUSE 100
3340 INPUT FLASH 1;"Che incremen
to in percentuale ";A
3345 IF A>0 THEN LET Z=Z+A
3350 LET SELLPRICE=INT (100*(SEL
LPRICE+A*SELLPRICE/100))/100
3360 PAUSE 50
3370 PRINT ' FLASH 1; INK 3;"Ora
 si venderanno a $ ";SELLPRICE
3380 PAUSE 100
3390 RETURN
3990 REM ********************
4000 REM VENDITE
4005 BEEP .5,RND*50: BEEP .4,RND
*50
4010 PRINT FLASH 1; PAPER 5;'"Lo
```

```
 stock completo e' di ";STOCK;"
unita'"
4015 PAUSE 100
4020 PRINT FLASH 1; PAPER 3;'"At
tendere il rapporto vendite ..."
4025 PAUSE 300
4030 CLS
4040 LET A=INT (RND*STOCK/(Z/100
))+1
4045 PAPER RND*6: CLS : BORDER R
ND*6
4050 IF A>STOCK THEN GO TO 4040
4055 BORDER 7: PAPER 7: CLS
4060 PRINT '' INK 1;"Totale lamp
adine vendute:";A
4070 LET STOCK=STOCK-A
4080 PRINT '"Ricavo dalle vendit
e: $";A*SELLPRICE
4090 LET CAPITAL=CAPITAL+A*SELLP
RICE
4095 PAUSE 100
4100 RETURN
5000 REM **********************
5010 REM STAMPA
5020 FOR G=40 TO 50: BEEP .008,G
: BEEP .008,60-2*G: NEXT G
5022 CLS
5025 IF CAPITAL+STOCK<1 THEN GO
TO 8000: REM BANCAROTTA
5027 IF CAPITAL+STOCK>999999 THE
N PRINT FLASH 1; BRIGHT 1; INK 2
;"Hai fatto un milione!!": PAUSE
 500: GO TO 8050
5030 PRINT INK2; FLASH 1;"RAPPOR
TO DELLA SETTIMANA NUMERO: ";WEE
K
5040 PRINT ' INK 2;"Capitale dis
ponibile: $";CAPITAL
5050 PRINT ' INK 1;"I magazzini
contengono ";STOCK;" lampadine",
"per un valore di $";STOCK*SELLP
RICE
5060 PRINT INK 2;"Ora si vendono
```

```
 a $";SELLPRICE;" ciascuna"
5070 PRINT INK 2;"e costano in p
roduzione $";COST
5080 PRINT INK 7; PAPER 1;"Il pe
rsonale e' di ";WORKFORCE;" oper
ai"
5090 PRINT PAPER 1; INK7;"La lor
o paga e' di $";WAGE;" ciascuno"
,"il totale settimanale e' di $"
;WAGE*WORKFORCE
5100 PRINT INK 2;"Ognuno di loro
 puo' fare ";PRODUCE,"lampadine
con una resa totale di ";PRODUCE
*WORKFORCE
5120 RETURN
5130 INPUT "Quante lampadive vuo
i produrre? ";MAKE
5135 IF MAKE=0 THEN RETURN
5140 IF MAKE*COST>CAPITAL THEN P
RINT INK 2; FLASH 1;"MANCANZA DI
 DENARO": GO TO 5130
5150 IF MAKE>PRODUCE*WORKFORCE T
HEN PRINT INK 4; FLASH 1;"MANCAN
ZA DI PERSONALE": GO TO 5130
5160 PRINT AT 0,0; FLASH 1; INK
1;"L'obbiettivo della settimana
";WEEK;" e' ";MAKE
5170 LET MAKE=MAKE-INT (RND*MAKE
/5*(Z/100))
5180 PAUSE 100
5190 PRINT INK 3; FLASH 1;AT 0,0
;"La produzione totale della set
timana ";WEEK;" e' stata ";MAKE
5200 LET STOCK=STOCK+MAKE
5210 LET CAPITAL=CAPITAL-COST*MA
KE
5220 PAUSE 50
5300 RETURN
5310 REM ********************
6000 REM ***PERSONALE***
6010 INPUT "Quanti operai vuoi a
ssumere ";A
6020 LET WORKFORCE=WORKFORCE+A
```

```
6030 PRINT AT 0,0; FLASH 1; INK
1;"  Personale disponibile: ";WO
RKFORCE
6035 PAUSE 100: GO SUB 5000
6037 IF A>0 THEN RETURN
6040 INPUT "Quanti operai vuoi "
;FLASH 1; INK 4;"licenziare? ";A
6042 IF A=0 THEN GO TO 6090
6045 IF A>WORKFORCE THEN GO TO 6
040
6050 LET A=INT (RND*A+1)
6060 PAUSE 100
6070 PRINT FLASH 1; BRIGHT 1; IN
K 6; PAPER 2,'''"I sindacati ti
permettono","di licenziare ";A;"
 operai"
6080 LET WORKFORCE=WORKFORCE-A
6090 PAUSE 100
6100 RETURN
7990 REM ********************
8000 REM BANCAROTTA
8010 PRINT '''TAB 8; FLASH 1; IN
K 2;"BANCAROTTA!!!!"
8020 PRINT '''"Comunque sei rius
cito a dirigere la fabbrica per
"; FLASH 1; INK 1;WEEK; FLASH 0;
 INK 0;" settimane."
8050 PRINT ''"Introduci -S- per
un'altra prova,o","-N- per finir
e"
8055 LET A$=INKEY$
8060 IF A$="" THEN GO TO 8055
8070 IF A$="S" OR A$="s" THEN RU
N
8080 STOP
9000 REM ************************
9010 REM VARIABILI
9020 REM ************************
9030 LET CAPITAL=500+INT (RND*50
0)
9040 LET STOCK=100+INT (RND*50)
9050 LET SELLPRICE=10+INT (RND*5
)
```

```
9060 LET COST=2+INT (RND*5)
9070 IF COST>SELLPRICE THEN GO T
O 9050
9080 LET WORKFORCE=7+INT (RND*10
)
9090 LET WAGE=12+INT (RND*SELLPR
ICE*5)
9100 LET PRODUCE=5+INT (RND*6)
9110 LET WEEK=0
9120 REM Z e' il fattore di
          resistenza alle vendite
9130 LET Z=1
9140 PAPER 7: CLS : BORDER 7: IN
K 0
9500 RETURN
```

Inizieremo con il ciclo dalla linea 40 alla 180:

- GO SUB 9000. Questa routine inizializza le variabili (i valori iniziali sono casuali e variano ogni volta): vengono stabilizzate quantità come il numero di persone che lavorano per voi, il loro stipendio iniziale, il prezzo di vendita delle lampadine e così via.
- La linea 50 incrementa la variabile WEEK
- La routine che inizia alla linea 5000 è richiamata diverse volte durante il ciclo principale (dalle linee 100, 120, 140) per mostrarvi i cambiamenti verificatisi durante la settimana.
- Le altre routine eseguono quanto indicato nelle frasi REM: la routine PERSONALE (licenziamenti-assunzioni) inizia alla linea 6000; la routine PRODUZIONE inizia alla 5130 e quella vendite alla 4000.
- La routine finale IMPREVEDIBILI (dalla linea 3000) è eseguita ogni 'settimana', ma alcuni o tutti i suoi compiti possono essere saltati. Gli IMPREVEDIBILI (che includono domande di aumento del sindacato e inondazioni che distruggono parte della merce contenuta nel vostro magazzino) sono progettati per rendervi la vita difficile, ma non sono arbitrari per evitare di eliminare completamente il peso della vostra abilità nella conduzione della fabbrica.

Per scrivere una simulazione dovete prima di tutto decidere cosa volete simulare e quali variabili sono necessarie al vostro programma. Fatto ciò dovete costruire il ciclo principale di controllo del programma. Il mio passo successivo, che potreste

seguire anche voi, è stato creare le stampe standard (in questo caso dalla linea 5000) anche prima di fare parti del programma che assegnavano o modificavano le variabili. Scoprirete che decidere prima le stampe standard vi aiuta a stabilire quali variabili vi occorrono.

Dato che le stampe sono la cosa più importante di questo programma sarà bene esaminarle per prime.

Le stampe appaiono durante il programma con diversi colori e talvolta sono lampeggianti per differenziare le diverse componenti e rendere più semplice la lettura. Quando fate girare il programma per la prima volta, vedrete i parametri iniziali che il caso ha scelto per voi e poi il programma esegue la routine PERSONALE, passando il controllo alla linea 6000. "Quanti operai vuoi assumere?" è la domanda che vi viene posta. Se decidete di assumerne, il loro numero sarà aggiunto al totale della forza lavoro; poi verranno stampate nuovamente le informazioni standard e potrete vedere come sono cambiate le condizioni del gioco. Si prosegue quindi con la successiva sezione del programma per determinare quante lampadine volete produrre.

Se invece decidete di non assumere personale vi verrà chiesto "Quanti operai vuoi licenziare?". È a questo punto che il sindacato fa la sua apparizione e decide di quante persone potete liberarvi, vengono quindi stampate nuovamente le informazioni standard.

Avendo risolto la situazione del personale, dovete ora fronteggiare le decisioni sulla produzione (con la routine che inizia alla linea 5130 - PRODUZIONE). "Quante lampadine vuoi produrre?". Se introducete zero il programma ritorna alla stampa principale e prosegue con la routine VENDITE, dove viene venduta parte della vostra merce. Se invece decidete un obiettivo di produzione, il programma lo metterà in relazione con:

(a) Il numero di persone che stanno lavorando per voi in questa settimana; tenete a mente che ogni persona ha una produzione limitata.

(b) Il costo della materia prima necessaria per ogni lampadina.

Se non avete abbastanza personale comparirà il messaggio MANCANZA DI PERSONALE, vi verrà quindi richiesto un altro obiettivo. Analogamente MANCANZA DI DENARO, comparirà sullo schermo se l'obiettivo richiede più materia prima di quanto potete permettervi. Quando l'obiettivo della simulazione è stato accettato verrà stampato in alto sullo schermo "L'obiettivo della settimana 4 è 92" o qualunque altro numero relativo. Dopo una breve pausa comparirà il messaggio "La produzione totale della settimana 4 è stata 86" o qualunque altro numero di lampadine prodotte.

Istruzioni troppo esplicite diminuirebbero il piacere di giocare con questa simulazione, così mi fermerò qui. Dopo tutto voglio lasciarvi ancora qualche sorpresa per quando farete girare il programma. Se avete seguito le spiegazioni fornite per gli altri giochi dovreste poter affrontare il programma senza troppi problemi.

Questo programma è stato incluso nel libro perchè le simulazioni sono uno dei campi in cui i computer si rivelano più utili. Creare le vostre simulazioni, una indica-

zione può essere simulare una stazione spaziale che produca frittelle, vi aiuterà a capire quanto è difficile creare dei modelli per simulare sul computer situazioni della vita reale. Ad esempio una vera fabbrica di lampadine deve tener conto di molti altri fattori quali la possibilità che i lavoratori restino feriti sul lavoro oppure non producano perchè sono ammalati, o lavorino lentamente perchè fa caldo, o perchè è il compleanno di qualcuno, o perchè è venerdì o perchè mancano pochi giorni a Natale e così via.

Potrebbero esserci dei ritardi nella consegna delle materie prime oppure potrebbe rompersi un'apparecchiatura di primaria importanza. Ci potrebbe essere un incendio o magari un blackout, oppure i sindacati potrebbero proclamare uno sciopero. Imparerete presto le difficoltà di simulare con troppa precisione la realtà.

Siamo così arrivati alla fine della sezione dei giochi. È stata la più ampia del libro perchè ritengo che scrivere e usare programmi di giochi sia il modo più sicuro (o quanto meno più piacevole) per migliorare la propria capacità di programmazione. Oltre tutto i giochi forniscono anche qualche spunto per usi più 'seri'. Spero che il materiale di questo capitolo vi aiuti a raggiungere entrambi gli scopi. Infine ecco alcuni dei libri che potreste trovare interessanti.

SUGGERIMENTI PER ULTERIORI LETTURE:

- *"BASIC Computer Games"* – Ahl David (ed.) (Creative Computing Press, USA 1980)
- *"What To Do After You Hit RETURN"* – (Hayden Book Company Inc., USA 1981)
- *"26 BASIC Programs For Your Micro"* – Daines Derrick (Newnes Technical Books/Butterworth & Co., 1982)
- *"67 Ready-To-Run Programs In BASIC"* – Watson Wm Scot (TAB Books Inc., USA 1981)
- *"Inside BASIC Games"* – Mateosian Richard (SYBEX Inc., USA 1981)
- *"Computer Programs That Work!"* – Lee J.D., Beech G., Lee T.D. (Sigma Technical Press, Wolverhampton 1980)
- *"Producing Computer Poetry"* – Chisman Margaret; articolo originale stampato in Creative Computing, ora disponibile (pagine 106-107) in The Best Of Creative Computing vol. 2 Ahl David (ed.) (Creative Computing Press, USA)

Qualche proposta per i vostri giochi:

- *"The Complete Book Of Indoor Games"* – Arnold Peter (ed.) (Hamlyn, 1981)
- *"The Complete Home Entertainer"* – Brandreth Gyles (Robert Hale, 1981)

CAPITOLO 7

GRAFICA TRIDIMENSIONALE

I film di fantascienza come 'Guerre Stellari' mostrano computer che producono immagini tridimensionali e le manipolano in tempo reale. Questo programma per lo Spectrum non può riprodurre immagini tridimensionali di un certo tipo e consentirvi poi di controllarle come se fossero oggetti reali, per mezzo di rotazioni.

Il programma vi permette di disegnare figure usando linee rette di lunghezza voluta. La spiegazione di come usare il programma può sembrare un po' complessa, ma se prima di cominciare inserite il programma e poi seguite attentamente le spiegazioni dovreste raggiungere un buon controllo del programma e riuscire a riprodurre tranquillamente immagini tridimensionali.

Prima di tutto introducete il programma, poi tornate al libro per la spiegazione su come usarlo.

```
  1 REM ************************
  2 REM       GRAFICA 3-D
  3 REM
  4 REM ************************
  5 DIM S$(255)
  6
 10 GO SUB 5000
 11 GO SUB 1000
 15 CLS
 20 GO SUB 7000
 35 IF P=1 THEN GO TO 200
 36
 40 LET R$=INKEY$: IF R$="" THE
N GO TO 40
 41 LET S$(A)=R$
 42 LET A=A+1
 43 IF A=255 THEN PRINT "BUFFER
 PIENO": STOP
```

```
  44 IF R$="E" THEN GO TO 10
  50 GO SUB 3000
  60 GO SUB 2000
  70 DRAW C-PEEK (23677),D-PEEK
(23678)
  80 GO TO 40
  90
 200 GO SUB 6000
 205 LET R$=INKEY$: IF R$="" THE
N GO TO 205
 210 GO TO 10
 220
1000 LET S=L*L+M*M
1010 LET T=S+N*N
1020 LET Q=SQR (T)
1030 LET H=SQR (S)
1040 RETURN
2000 LET O=T-U*L-V*M-W*N
2010 LET C=T*(V*L-U*M)*4/(H*O)+1
28
2020 LET D=96+3*Q*(W*S-N*(U*L+V*
M))/(H*O)
2286 RETURN
2287
3000 IF R$="7" THEN LET W=W+G
3010 IF R$="6" THEN LET W=W-G
3020 IF R$="5" THEN LET U=U-G
3030 IF R$="8" THEN LET U=U+G
3040 IF R$="I" THEN LET V=V-G
3050 IF R$="A" THEN LET V=V+G
3060 RETURN
3999
5000 CLS
5010 INPUT "SCALA?";G
5020 INPUT "COORDINATA X?";L
5030 INPUT "COORDINATA Y?";M
5040 INPUT "COORDINATA Z?";N
5050 LET A=1
5061 PRINT "'G'-GET"'"'P'-PUT"'"
'R'-RANDOM"
5070 LET R$=INKEY$: IF R$="" THE
N GO TO 5070
5080 IF R$="G" THEN LET P=1: GO
```

```
TO 5110
5090 IF R$="P" THEN LET P=0: GO
TO 5110
5093 IF R$="R" THEN GO TO 8000
5100 GO TO 5070
5110 RETURN
5999
6000 GO SUB 7000
6010 LET R$=S$(A)
6020 GO SUB 3000
6030 GO SUB 2000
6040 DRAW C-PEEK (23677),D-PEEK
(23678)
6050 LET A=A+1
6060 IF S$(A)<>"E" AND A<>255 TH
EN GO TO 6010
6070 RETURN
6999
7000 LET W=0: LET U=0: LET V=0:
GO SUB 2000
7010 PLOT C,D
7020 RETURN
7999
8000 LET A=1: LET G=20: LET L=RN
D*100: LET M=RND*100: LET N=RND*
100
8003 CLS
8010 GO SUB 6000
5020 PAUSE 50
8030 GO TO 8000
```

Assicuratevi che il CAPS LOCK sia inserito (tenete premuto CAPS SHIFT e premete il tasto 2) prima di fare eseguire il programma. Quando lo farete girare apparirà per prima cosa la domanda SCALA? in basso sullo schermo. Il programma vi chiede di introdurre la dimensione delle linee che comporranno la figura da disegnare. Per avere un'idea della scala che usa il programma inserite 20, poi premete ENTER.

La domanda successiva è COORDINATA X? Per questa prima prova inserite 100, poi 150 e 20 rispettivamente per le coordinate Y e Z.

Lo schermo si dovrebbe cancellare per mostrarvi il seguente menù:

'I' – INSERIRE
'R' – RICHIAMARE
'C' – CASUALE

Queste sono le tre scelte che vi vengono proposte. Inizierete con l'opzione INSERIRE, che potete selezionare con il tasto 'I'. Se non accade nulla quando premete 'I' probabilmente non avete inserito il CAPS LOCK.

Lo schermo si cancellerà ancora lasciando un piccolo punto al centro. Questo è il punto iniziale della figura da disegnare.

Il programma accetta ora sei comandi differenti:

'7' – Alto
'6' – Basso
'5' – Sinistra
'8' – Destra
'I' – Indietro
'A' – Avanti

Dovete soltanto premere il tasto corrispondente alla direzione in cui volete muovervi. Non tenete i tasti premuti troppo a lungo o finirete per disegnare segmenti più lunghi di quanto intendevate fare. Giocate un po' con il programma, premendo i tasti direzionali e poi tornate al libro.

Ora provate questo semplice disegno:

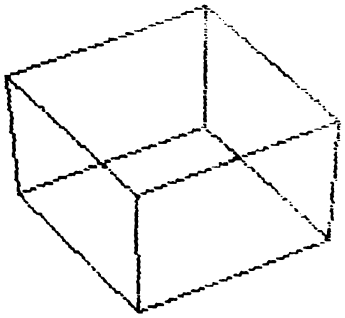


Figura 1

Cancellate lo schermo premendo 'F' (per Fine) oppure ripartendo con RUN. Entrambi i metodi vi riportano alla domanda SCALA? Usate gli stessi dati introdotti prima e inserite 'I' quando vi verrà proposto il menù.

Premete '7'. Disegnate una linea dal punto iniziale a circa tre quarti dello schermo. Premete '5' per spostarvi a sinistra e poi '6' per spostarvi verso il basso. Ora dovreste avere sullo schermo qualcosa di simile a una U rovesciata.
Unite i due terminali in modo da ottenere un quadrato con il tasto '8'.

Ora premete il tasto 'A' e ripetete la procedura dal punto in cui vi trovate. Fatto ciò dovreste accorgervi che vi mancano solo tre linee per avere un cubo tridimensionale, come quello in figura 1.

Per raggiungere i punti da cui partire per disegnare le linee mancanti, abbiamo bisogno di passare su alcune linee già tracciate. La sequenza di tasti richiesta è:

7, I, A, 5, I A, 6, I, A e 8.

Se osservate attentamente lo schermo mentre inserite i comandi, potrete rendervi conto di come lo Spectrum esegue le vostre istruzioni. A questo punto premere 'F' e ripetere l'intera procedura diverse volte potrebbe essere utile per imparare a disegnare un cubo senza commettere errori.

Ora, per continuare i nostri studi di grafica tridimensionaleg usate 'F', per ritornare al messaggio SCALA? e rispondete come segue:

```
SCALA? 10
COORDINATA X ? 100
COORDINATA Y ? 150
COORDINATA Z ? 200
```

Quindi scegliete l'opzione RICHIAMARE del menù.

Troverete che il vostro cubo viene disegnato automaticamente, ma in scala ridotta. Se invece non appare niente, vuol dire che siete usciti dal programma e avete usato RUN, invece di terminare il disegno con il comando 'F'. Il programma conserva la sequenza di tasti che avete usato per disegnare l'ultima figura, ma quando fate girare il programma per la prima volta non avete figure memorizzate. Potete provare a usare 'F' e RICHIAMARE per ridisegnare il cubo diverse volte, dando parametri diversi alle domande del programma.

Come funziona?

Il programma disegna sempre la vostra figura partendo dalle coordinate tridimensionali 0, 0, 0. Le risposte alle domande delle coordinate, dicono al computer da che punto nello spazio volete cominciare a disegnare. Il menù (INSERIRE/RICHIAMARE/CASUALE) vi chiede se la figura da disegnare deve essere inserita con una sequenza di tasti (I) o si trova memorizzata nel programma ed è quindi l'ultima figura disegnata (R). L'opzione 'C' fa eseguire una sequenza senza fine che disegna la figura partendo da coordinate casuali e dopo una pausa la ridisegna usando altre coordinate casuali. Ovviamente l'opzione 'C' come l'opzione 'R', richiede che ci sia

già una immagine in memoria. La figura due mostra un disegno che potete provare a duplicare.

Le figure da tre a sei mostrano un'altra immagine, più semplice, disegnata partendo da coordinate diverse. La figura tre è il disegno originale. La figura quattro è lo stesso disegno realizzato con una SCALA più piccola. La figura cinque è sempre lo stesso disegno realizzato da un altro lato. La figura sei riprende il disegno dal lato inferiore. Siccome il programma produce solo i lati principali dell'immagine, è difficile stabilire la direzione in cui è rivolta la figura sei. Troverete che le figure complesse vengono meglio se disegnate con una piccola SCALA, ad esempio 10, per permettervi di introdurre più dettagli nel disegno.

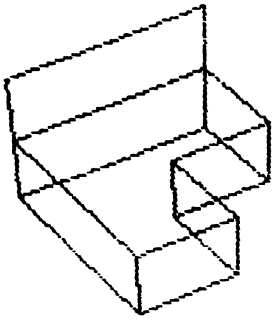


Figura 2:

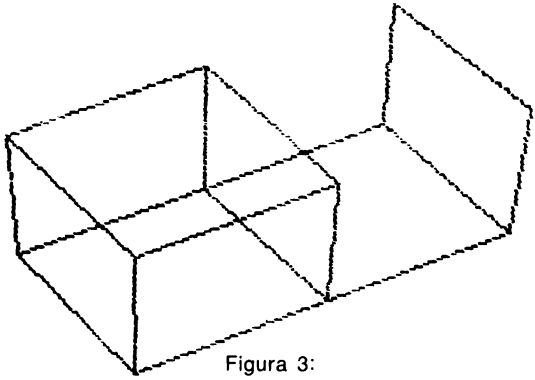


Figura 3:

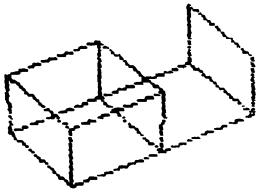


Figura 4:

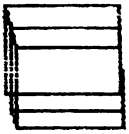


Figura 5:

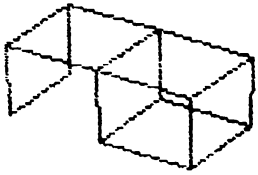


Figura 6:

Il programma è costituito da diverse routine di cui parlerò subito per tornare in seguito sulla sezione che va dalla linea 1 alla 210.

Routine 1000 — Nuove coordinate. Questa routine legge le nuove coordinate in L, M e N e usa questi parametri per assegnare il valore alle variabili S, T, Q e H. Queste variabili sono richieste in seguito dall'algoritmo di disegno.

Routine 2000 — Elabora le coordinate bidimensionali. Questa routine restituisce in C e D le coordinate di schermo corrispondenti al punto U,V,W. Essa usa L, M e N e tutte le costanti definite dalla routine 1000. Le costanti 128 e 96 alle linee 2010 e 2020 servono a centrare l'immagine, assumendo che il punto 0,0,0 si trovi al centro dello schermo.

Routine 3000 — Modifica le coordinate. Le coordinate attuali U,V,W sono incrementate o decrementate del valore G per ogni tasto premuto, quando vengono usati i tasti di direzione. Per fare ciò la routine memorizza il tasto premuto in R$.

Routine 5000 — Menù. Questa routine svolte il dialogo con voi. Le linee da 5010 a 5040 pongono le ormai familiari domande, la linea 5061 stampa il menù e la 5070 legge la scelta relativa. La variabile A alla linea 5050 è il puntatore di stringa e serve quando tutti i tasti sono stati premuti e viene richiesta l'opzione R, o l'opzione C. Le linee da 5080 a 5093 decodificano il tasto premuto. Se era stata selezionata l'opzione R, P è vero, se viene selezionata l'opzione I, P è falso. Se viene richiesta l'opzione casuale viene richiamata la routine alla linea 8000.

Routine 6000 — Disegno. Questa routine disegna automaticamente la figura memorizzata in S$. Viene usata per l'opzione R.

Linea 6000: chiama la routine 7000 che inizializza il punto al centro dello schermo.

Linea 6010: legge il prossimo carattere da S$ e lo memorizza in R$, per la chiamata alla routine 3000 della linea 6020.

Linea 6030: elabora le nuove coordinate per il disegno.

Linea 6040: esegue una DRAW 'assoluta' a queste nuove coordinate. Le due PEEK servono per trovare i valori attuali X e Y delle coordinate dello Spectrum; queste sono sottratte dalle coordinate assolute per ottenere un movimento assoluto piuttosto che uno relativo. Questa tecnica può essere applicata anche in altri programmi, dove la forma normale di DRAW risulta impropria.

Linea 6050: incrementa il puntatore di S$, per esaminare il prossimo carattere.

Linea 6060: se il carattere è una 'F' siamo alla fine del disegno, quindi viene eseguita una RETURN. La stessa cosa succede se il puntatore A raggiunge il valore 255.

Linea 6070: torna indietro per gli altri caratteri di S$.

Routine 7000 — Pone le coordinate attuali a 0,0,0 e vi stampa un punto.

Routine 8000 — Stabilisce i valori per le variabili A, G (la scala), L, M e N poi ese-

gue il disegno (linea 8010), esegue una pausa poi torna indietro ed effettua il disegno partendo da coordinate differenti.

Linea 5: Dimensiona S$.
Linea 10: Chiama la routine 5000.
Linea 11: Chiama la routine 1000.
Linea 15: Cancella lo schermo.
Linea 20: Chiama la routine 7000.
Linea 35: Manda il controllo alla linea 200 se viene usata l'opzione 'R'.
Linea 40: Legge un tasto per la direzione.
Linea 41: Memorizza il tasto in S$.
Linea 42: Incrementa il puntatore A.
Linea 43: Ferma il programma se sono stati premuti più di 255 tasti.
Linea 44: Ritorna al menù se viene premuto il tasto 'F'.
Linea 50: Esegue il comando.
Linea 60: Decodifica le coordinate.
Linea 70: Disegna una linea fino alla nuova posizione (vedi descrizione per la linea 6040).
Linea 80: Rimanda il controllo alla linea 40, per leggere un altro tasto.
Linea 200: Chiama la routine 6000 per l'opzione RICHIAMARE.
Linea 205: Attende che venga premuto un tasto.
Linea 210: Fa tornare il programma al menù.

CAPITOLO 8

INTRODUZIONE AL CODICE MACCHINA

Ricorderete che all'inizio del libro abbiamo parlato dei linguaggi per i computer e abbiamo detto che sono divisi in livelli, con linguaggi ad alto livello come il BASIC, abbastanza vicini all'Inglese e linguaggi a basso livello più vicini al linguaggio reale dei microprocessori, la parte pensante del vostro computer. Il linguaggio usato dal microprocessore (nello Spectrum è uno Z80) è un linguaggio chiamato codice macchina. Quando parlate al vostro Spectrum in BASIC, un parte del computer deve impiegare del tempo per tradurre il messaggio BASIC in codice macchina, così il microprocessore può capirlo. Ma questa traduzione richiede un certo tempo, ecco perchè il BASIC è un linguaggio piuttosto lento. Come dimostreremo ora il codice macchina è molto più veloce.

Scrivete e fate eseguire questo programma:

```
 0 CLEAR 32000
20 RESTORE
30 FOR a=0 TO 15: READ x: POKE 32000+a,x: NEXT a
40 DATA 33,255,63,01,01,24,22,255,35,11,120,177,
   200,114,24,248
50 CLS: PAUSE 20: RANDOMIZE USR 32000: PAUSE 20: GO TO 50
```

Questo dimostra come può essere riempito lo schermo. Ora confrontatelo con l'equivalente BASIC:

```
10 FOR a=1 TO 704: PRINT "*";: NEXT a
```

NOTA: NON DOVETE GIUDICARE LA LUNGHEZZA DI QUESTI DUE PROGRAMMI, PERCHÈ IL PRIMO PROGRAMMA È IN REALTÀ MOLTO PIÙ LUNGO DEL NECESSARIO (COME VEDREMO PIÙ TARDI).

Il computer può comprendere solo lunghe serie di cifre. Per esempio, una istruzione dello Z80 è 00111101, che per la maggior parte di voi non significherà niente. (Notate che lo Z80 accetta codici di otto cifre, il che significa che il numero massimo che può trattare è 11111111 o 255 in base dieci, il sistema di numerazione che usiamo quotidianamente).

Questa stringa di uno e zero è un numero binario. Un numero nel nostro sistema normale, in base dieci, ha cifre comprese tra zero e nove; un numero binario, in base due, ha solo le cifre zero e uno. Se osserviamo un numero in base dieci, la cifra più a destra rappresenta le unità, la cifra successiva rappresenta le decine (o dieci elevato a uno); la terza cifra rappresenta le centinaia (o dieci elevato a due), e così via. Ad esempio: 101 è uguale a 1 unità, 0 × 10 e 1 × 100.

La base due opera sullo stesso principio. La prima cifra sulla destra rappresenta ancora le unità (due elevato a zero), la cifra successiva è due elevato a uno e la terza cifra è due elevato a due. Per esempio, in binario 101 equivale a 1 unità, 0 × 12, 1 × 4 che è 5 in base 10. Così se prendiamo 11111111, che è il numero più grande che lo Z80 può trattare, apparirà come segue:

```
  1 per   1
+1 per   2
+1 per   4
+1 per   8
+1 per  16
+1 per  32
+1 per  64
+1 per 128
= 255
```

Mentre siamo sull'argomento dei numeri binari, potrebbe essere utile chiarire una questione che può causare molti problemi. Quando sentiamo parlare di computer udiamo spesso i termini 'bit' e 'byte'. Un bit è una delle cifre del numero binario. Per esempio nel numero 00111010 ci sono otto bit. Possiamo parlare di numeri binari riferendoci ai bit che li costituiscono. Il bit 0 (la cifra più a destra) è nel nostro esempio 0 e il bit 3 (la quarta cifra da destra) è uno. Byte è la parola per un numero con otto bit, così 10101101 è un byte. Un bit ha un valore di 1 e il massimo valore di un byte e 255 in base 10. Useremo codici a otto cifre invece che a sei o a dieci cifre perchè il microprocessore Z80 è progettato come un microprocessore a otto bit, il che significa che può trattare dati di otto bit, nè di più nè di meno. Così, per quanto concerne i dati, siamo ristretti ad un numero massimo di 255. Potete verificarlo scrivendo POKE 100,365. Il computer rifiuterà questa istruzione e risponderà con un messaggio di errore.

Esaminiamo ora il comando che diamo al computer quando usiamo l'istruzione POKE. Iniziamo con l'indirizzo di un particolare byte della memoria e proseguiamo con il numero che vogliamo memorizzaci. Possiamo pensare la memoria del computer come ad una serie di piccole scatole ognuna delle quali ha un particolare numero binario (l'indirizzo della scatola) e ne contiene un altro (il suo valore).

Come abbiamo detto prima con un solo byte, otto bit il numero assimo che possiamo avere nello Spectrum è 11111111 o 255 in base 10. Questo numero può anche andarci bene, se si tratta di un contenuto, ma immaginiamo di volere che l'indirizzo

di ogni scatola abbia un numero compreso tra 0 e 255. Ciò che dobbiamo fare è aggiungere il nome di una strada all'indirizzo della scatola, per lo Spectrum questa strada sarà ancora un numero binario. Esaminiamo il primo indirizzo diciamo che il nome della strada è 1, qui ci saranno i primi 256 indirizzi (0-255). L'indirizzo completo apparirà di questo tipo:

```
00010110          01011010
nome della strada numero della scatola
```

Come potete vedere poichè ci sono 256 nomi di strade e 256 numeri di scatole abbiamo 256 x 256 = 65536 indirizzi. Tecnicamente ciò è noto come un numero a 16 bit e presenta molti vantaggi, principalmente perchè possiamo pensare ad esso come ad un numro lungo 16 bit. Per esempio se abbiamo un numero come 10101101 e vogliamo aggiungere 10010001 troviamo che usciamo dalle otto cifre.

```
  10101101
  10010001
 ========
  1:00111110
 ========
OVERFLOW    numero a 8 bit
```

Ma se usiamo registri a 16 bit non abbiamo più problemi:

```
 00000000    10101101
 00000000    10010001
================
 00000001    00111110
================
```

Questo significa che possiamo tranquillamente moltiplicare due numeri sapendo che il risultato dovrà essere maggiore di 65535 prima di superare il numero di cifre a disposizione.

Tornando agli indirizzi possiamo capire ora, sapendo che lo Z80 usa indirizzi a 16 bit, perchè il numero massimo di locazioni è 65535. Se lo dividete per 1024 (1 kilobyte o 1K) ottenete 64K che è la quantità massima di memoria disponibile per un computer con lo Z80. Notate che questo è vero anche sullo ZX Spectrum che ha una memoria massima di 16K ROM e 48K RAM (16 + 48 = 64K).

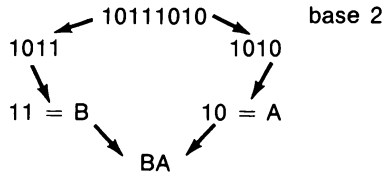
RAM sta per Random Access Memory (memoria ad accesso casuale) e ROM sta per Read Only Memory (memoria a sola lettura). Se consideriamo una locazione di memoria come una casa, possiamo dire che con la RAM potete guardare attraverso le finestre e vedere il contenuto di ciascuna, aprire la porta e modificarlo. Con la

ROM potete solo guardare attraverso le finestre e 'leggere', ma la porta è chiusa a chiave così non potete entrare e modificare il contenuto.

Trattare numeri binari è troppo complicato per gli usi normali. Ecco perchè interviene il terzo e più importante sistema di numerazione, quello in base 16. Con questa base un numero binario a otto cifre può essere scritto con due cifre in base sedici, cioè con due numeri esadecimali. Ricorderete che un numero in base due è composto dalle cifre 0 e 1 e uno in base 10 è scritto con cifre comprese tra 0 e 9; così un numero in base dovrà essere scritto con cifre comprese tra 0 e 15. Ma poichè non abbiamo cifre maggiori di 9 usiamo anche le prime sei lettere dell'alfabeto:

10 = A
11 = B
12 = C
13 = D
14 = E
15 = F

Il modo più semplice di convertire un numero binario a 8 bit in un numero esadecimale (base 16) è separare gli 8 bit in due gruppi di 4 bit e convertire in un singolo carattere e poi combinare i due carattri. Per esempio:



Con questo metodo possiamo convertire qualunque valore di un byte in due cifre esadecimali. Questo è il sistema più conveniente e più comune per rappresentare un singolo byte.

La tabella che segue mostra gli equivalenti binario, decimale ed esadecimale.

Decimale	Esadecimale	Binario
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000

Decimale	Esadecimale	Binario
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Ora che abbiamo capito la base matematica del codice macchina (i noiosissimi bit) possiamo passare alla programmazione vera e propria.

Lo Z80 è in grado di comprendere solo codici numerici invece di istruzioni reali. Ciò che dobbiamo fare è scrivere il nostro programma usando una istruzione di tipo mnemonico e poi convertirlo in un numero o un gruppo di numeri che possano essere caricati in memoria in forma decimale, con la istruzione POKE ed eseguiti con l'istruzione USR.

Provate ad esempio a introdurrre nel vostro Spectrum:

```
CLEAR 32000
```

che cancella la memoria dalla locazione 32000 alla fine della RAM (32767 nella versione a 16K e 65535 nella versione a 48K). Questa istruzione protegge la vostra routine dalla possibilità che un programma BASIC invada l'area di memoria in cui è scritta e la cancelli, oppure che venga distrutta dall'istruzione NEW. 32000 è il primo indirizzo in cui potete scrivere il vostro codice macchina. Il prossimo problema è introdurre il codice macchina in questa area di RAM. Per fare questo si può usare l'istruzione POKE.

Provate questo:

```
POKE 32000,1 (premete ENTER)
POKE 32001,0
POKE 32002,0
POKE 32003,201
```

Ora scrivete PRINT USR 32000 ed esso stamperà 0. Introducendo con POKE i numeri si possono caricare al massimo una trentina di codici, poi il procedimento diventa molto noioso. Ci sono due modi per semplificare questo compito:

(1) Usando un ciclo FOR/NEXT:

```
10 FOR a=32000 TO 32003: INPUT x: POKE a,x: NEXT a
```

(2) Usando una frase DATA:

```
 5 RESTORE
10 FOR a=32000 TO 32003: REAd x: POKE a,x: NEXT a
20 DATA 1,0,0,201
```

Entrambi questi metodi hanno applicazioni particolari, ma io preferisco il secondo che fa uso della frase DATA, perchè è di più facile lettura e, se necessario, si può modificare.

Questo che segue è un programma per caricare il codice macchina nello Spectrum:

```
10  INPUT "Indirizzo di inizio?";start
20  CLEAR start
25  RESTORE
30  LET start=1+(PEEK 23730+256*PEEK 23731)
40  FOR a=start TO 65535 (usate 32767 nella versione 16K)
50  READ x: IF x=999 THEN STOP
60  POKE a,x
70  NEXT a
80  DATA ...introducete qui la vostra routine...
```

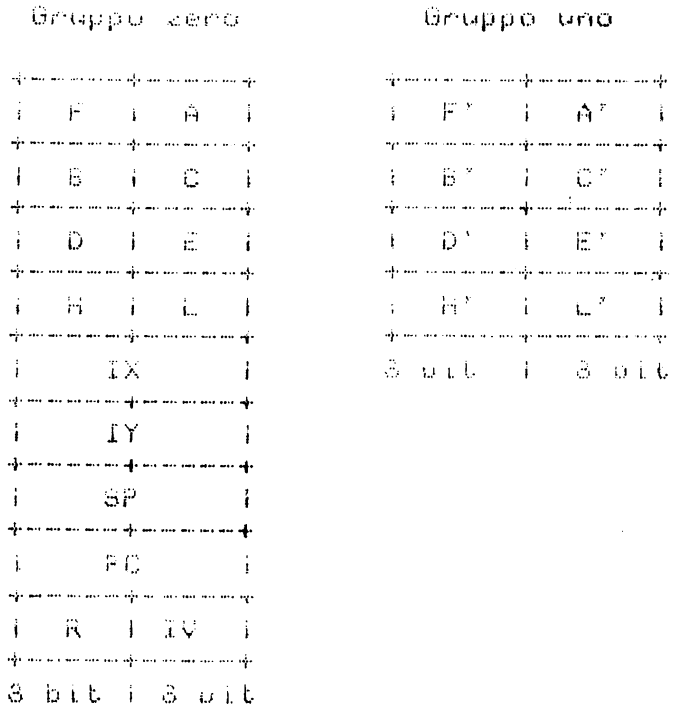
Esaminiamo il programma linea per linea per vedere come funziona: la linea 1P vi chiede l'indirizzo di partenza del programma. Abbiamo già discusso nell'indirizzo di partenza precedentemente. La linea 20 cancella la locazione di partenza e tutte le seguenti. La 25 fa uso dell'istruzione RESTORE per posizionare il puntatore DATA sul primo dei dati. La linea 30 riassegna alla variabile start il suo valore originale.

Questo si rende necessario perchè l'istruzione CLEAR cancella tutte le variabili. La linea 40 inizializza un ciclo che va dalla locazione start al massimo indirizzo possibile sullo Spectrum (65535 nella versione a 48K e 32767 in quella a 16K). La linea 50 legge il primo elemento della istruzione DATA e controlla se è uguale a 999, che è un codice illegale per l'istruzione POKE; in questo modo siamo in grado di identificare la fine dei dati. La linea 60 carica il valore appena letto dalla frase DATA (x) all'indirizzo tramite l'istruzione POKE. La linea 70 ripete il ciclo. La linea 80 contiene i valori decimali della vostra routine e termina con un 999. Non è necessario avere tutti i dati su un'unica frase DATA. Potete usare quante linee volete, compatibilmente con la memoria disponibile.

Così questa routine carica il codice macchina nella RAM. Per fare eseguire il programma dovete includere una linea PRINT USR start o RANDOMIZE USR start nel vostro programma. USR è un comando del BASIC (che potete ottenere dal tasto L), e sta per User SubRoutine (routine definita dall'utente).

Non ci sono variabili nel codice macchina. Esistono invece dei registri che possono essere caricati, letti o manipolati.

Ecco un semplice schema dei registri dello Z80:



Ogni registro definito con una lettera (B, C, B', C', D ecc) è a 8 bit, contiene quindi un singolo byte; tuttavia questi registri sono stati progettati per essere composti, in modo da ottenere registri a 16 bit. I registri del gruppo zero, eccetto F, possono essere utilizzati dall'utente. F è usato dal computer per alcune sue operazioni e di conseguenza non deve essere alterato.

Le coppie di registri IX e IY sono registri indice, di cui non si occuperemo. SP è chiamato Stack Pointer ed è usato anch'esso dal computer, tuttavia un programmatore esperto in codice macchina può usarlo per i suoi scopi.

PC è il Program Counter e contiene l'indirizzo della locazione di memoria dove è situata la prossima istruzione da eseguire. R è un altro dei registri usati dal computer e assume valori apparentemente casuali. Volendo si può modificare il contenuto di questo registro ma, considerato che il suo è uno scopo ben preciso, è preferibile lasciarlo invariato.

I registri del gruppo uno sono a vostra disposizione ma non è molto facile usarli. Per il momento concentreremo la nostra attenzione sui registri del gruppo zero, eccetto F; quindi A, B, C, D, E, H e L.

Come ho già detto questi sono registri a 8 bit, con una capacità massima di 255. Il codice macchina dello Z80 è stato scritto in modo da rendere semplice l'uso di que-

sti registri in coppie; combinando B con C, D con E e H con L si ottengono dei registri a 16 bit (BC, DE, HL). Come forse ricorderete questo ci consente per ogni registro a 16 bit una fascia di valori compresa tra 0 e 65535.

Dalla possibilità di combinare i registri ci spostiamo ora su un altro argomento interessante: come caricare in questi registri un determinato valore. Per caricare in un singolo registro un certo numero facciamo uso di un'istruzione del tipo: LD registro, numero.

Così, se vogliamo caricare nel registro A (A sta per Accumulatore) il valore 1, l'istruzione in codice macchina sarà:

```
LD A,01
```

Possiamo operare in questo modo con qualsiasi registro a 8 bit: A, B, C, D, E, H o L:

```
LD B,255
```

Il numero caricato in ogni registro deve essere compreso tra zero e 255. Se decidiamo di trattare D ed E in coppia (DE), possiamo lavorare nello stesso modo:

```
LD DE,1025
```

In questo caso però possiamo usare numeri compresi nella fascia tra 0 e 65535.

Dobbiamo ricordare che prima di poter introdurre la nostra istruzione nel computer, dobbiamo convertirla dalla forma LD B,255 in un codice decimale. Questa che segue è una tabella dei comandi LD e dei relativi codici; è un numero compreso tra zero e 255:

```
LD A,xxx          62 xxx
LD B,xxx          06 xxx
LD D,xxx          22 xxx
LD E,xxx          30 xxx
LD C,xxx          14 xxx
LD H,xxx          38 xxx
LD L,xxx          46 xxx
LD BC,xxx xxx     01 xxx xxx
LD DE,xxx xxx     17 xxx xxx
LD HL,xxx xxx     33 xxx xxx
```

Non solo potete caricare in un registro un dato numero, ma potete anche caricarvi il valore di un altro registro. Per esempio LD A,B significa: carica il contenuto del registro B nel registro A. Sfortunatamente non esistono istruzioni che consentano di

caricare una coppia di registri con il contenuto di un'altra. Se desiderate caricare HL con il contenuto di Bc, dovete scrivere:

```
LD H,B
LD L,C
```

Notate che in codice macchina non esistono numeri di linea. Questo significa che se volete eseguire un salto o un ciclo dovete saltare ad uno specifico indirizzo di memoria. Prima di esaminare i salti, ci sono altre due cose che è bene sappiate sui registri.

Se volete sottrarre un numero dal registro A potete usare la seguente istruzione:

```
SUB A,01
```

che significa LET A=A−1 o sottrai 1 dal contenuto di A. Potete anche sottrarre il contenuto di un altro registro dal contenuto di A:

```
SUB A,B
```

I codici di tutte queste istruzioni sono elencati nel manuale dello Spectrum (pagg. 183-188 della prima edizione). Potrete trovarli anche in molti altri libri che trattano del microprocessore Z80.

Dopo aver richiamato una routine in codice macchina, in altre parole dopo aver fatto eseguire al computer un'istruzione del tipo PRINT USR . . . o RANDOMIZE USR . . . . avete bisogno di un'istruzione in codice macchina che dica al computer di tornare al BASIC. È la stessa cosa che succedeva quando usavate un'istruzione GO SUB; avevate bisogno di un'istruzione RETURN per tornare al programma principale. Questa istruzione che dice al computer di tornare al BASIC è RET ed ha il codice decimale 201.

C'è un'altra cosa che vorrei puntualizzare prima di lasciarvi al remunerativo compito di scrivere routine in codice macchina. Se scrivete PRINT USR xxx, il numero che verrà stampato sarà il valore della coppia di registri BC dopo l'esecuzione della routine. Ad esempio, se caricate la seguente routine:

```
LD B,0
LD C,0
RET
```

che potete scrivere inserendo i codici 06,00,14,00,201,999 tramite il programma di caricamento che abbiamo visto precedentemente e la fate eseguire con l'istruzione PRINT USR 32000, otterrete in stampa il valore della coppia BC; esso sarà naturalmente zero (dato che la routine ha caricato zero in BC).

Ora provate a modificare il secondo e il quarto codice.

Questi sono i valori che vengono caricati in BC. Il valore di B è il numero di 256 contenuti nel numero assegnato a BC e il valore di C è il resto (compreso tra 0 e 255). Quando calcolate il valore da assegnare a una coppia di registri (LD HL, xxx xxx) è importante ricordare che il resto (o byte basso) viene caricato per primo, poi viene sommato il byte alto moltiplicato per 256. Per esempio se il byte alto è zero e il byte basso è 100, H verrà caricato con 0 e L con 100. Usando la notazione mnemonica scriveremo LD HL,000 100 che in decimale diventa 33 100 000. Notate che i due numeri devono essere scambiati prima di essere convertiti in codici.

Provate questa routine:

```
LD B,0      06 00
LD C,0      14 00
LD A,12     62 12
SUB A,6     214 06
LD C,A      79

RET 201
```

Che esegue questi compiti:

```
LD B,0      Carica 0 in B
LD C,0      Carica 0 in C
LD A,12     Carica 12 in A
SUB A,6     Sottrae 6 da A
LD C,A      Carica il valore di C in A
RET         Ritorna al BASIC
```

L'equivalente BASIC sarebbe:

```
LET B=0
LET C=0
LET A=12
LET A=A-6
LET C=A
RETURN
```

La versione BASIC richiede più di 60 byte di memoria. Il codice macchina utilizza solo 10 byte ed è molto più veloce.

Siamo così giunti alla fine di questa introduzione sul codice macchina. Naturalmente lo scopo di questo capitolo non era quello di farvi diventare degli esperti di codice macchina, ma semplicemente darvi una prima conoscenza dei problemi legati a questo modo di programmare, per fornirvi una base da cui partire.

SUGGERIMENTI PER ULTERIORI LETTURE:

- *"Z80 Instruction Handbook"* – Nat Wadsworth (Scelbi Publications, USA, 1978). Questo libretto spiega, in termini facilmente comprensibili a tutti, le possibilità del set di istruzioni dello Z80. Può servire come guida per tutte le istruzioni in codice macchina della CPU Z80; per ognuna vengono forniti l'istruzione mnemonica, il codice numerico e la spiegazione del funzionamento. Dice Mr Wadsworth: "È stato studiato per servire al dilettante, all'esperto o al professionista che abbia necessità di lavorare con una macchina basata sul microprocessore Z80".

- *"Z80 Software Gourmet Guide and Cookbook"* – Nat Wadsworth (Scelbi Pubblications, USA, 1979).
  Questo libro contiene una descrizione completa del set di istruzioni dello Z80 e una vasta gamma di routine che vi forniranno molte informazioni sulla programmazione dello Z80.

CAPITOLO 9

GUIDA A UNA MIGLIORE PROGRAMMAZIONE

Nella programmazione, come in molte aree del lavoro umano, è stata costruita una scala di valori che classifica le varie tecniche in "buono", "non molto buono", "pessimo". Quando avete cominciato a usare lo Spectrum, essere in grado di scrivere un programma funzionante era già di per sè un valido risultato. Ma dopo un certo tempo è probabile che abbiate incominciato a pensare che valeva la pena di provare a migliorare la vostra tecnica di programmazione. Così facendo, non solo otterrete dei programmi che saranno molto più semplici da correggere e modificare, ma avrete anche un lavoro in cui si potrà individuare subito la sezione che svolge un determinato compito. Se in un secondo tempo deciderete di modificare un programma, avrete meno problemi con un programma ben strutturato piuttosto che con una massa intricata di GO TO saltellanti e lunghe frasi IF THEN. Un programma ben scritto è anche più semplice da usare.

Comunque non esistono regole incise su tavole di pietra. Io non accetto nessuna regola incondizionatamente e vi suggerisco di fare altrettanto. L'uso della istruzione GO TO, ad esempio, è considerato come un cattivo modo di lavorare in programmazione; ma in un BASIC come quello implementato sullo Spectrum che non dispone di istruzioni quali REPEAT/UNTIL (ripeti fino a ...) o DO/WHILE (esegui mentre ...) capita spesso di non poterne fare a meno. Comunque, vale la pena di esaminare tutti i GO TO incondizionati che appaiono nel vostro programma per vedere se è possibile evitarli spostando un blocco di istruzioni da un punto all'altro del programma, oppure facendo uso di una routine o di un ciclo FOR/NEXT.

Un altro esempio di norma di buona programmazione che si può tranquillamente ignorare sullo Spectrum è quella che dice di non uscire da un ciclo con un salto, ma di lasciarlo sempre terminare. Su alcuni computer, come il BBC Microcomputer e l'Acorn Atom, è impossibile uscire da troppi cicli in un programma senza provocare un arresto del sistema, ma sullo Spectrum potete farlo quante volte volete senza alcun danno. E un ciclo FOR/NEXT incompleto è senza dubbio un peccato meno grave di un'infinità di IF X=3 THEN LET X=X+1: GO TO Y che spesso rappresenta l'unica alternativa all'uscita prematura da un ciclo.

Tenete quindi presente, mentre leggete il resto di questo capitolo, che le 'regole'

esposte sono più che altro dei suggerimenti. Leggetele e rifletteteci, ma sentitevi completamente liberi di ignorarle se ritenete che non siano valide per un particolare problema. Tuttavia nessuna rgola è stata inclusa senza un motivo ben preciso; quindi, prima di scartarle, provate a vedere se non vi possono essere di qualche aiuto.

Molti libri di programmazione suggeriscono di iniziare un programma tracciando un 'flowchart', cioè uno schema di rettangoli e rombi collegati che descrive il corso del programma quando viene eseguito. Sebbene non sia importante eseguire diagrammi molto elaborati, potreste trovare utile preparare un diagramma a blocchi, anche grossolano, di cosa deve fare il programma, prima di inserire le istruzioni nel computer.

Qualche volta tutto ciò di cui avete bisogno è una serie di parole chiave collegate da linee e frecce. Una cosa del genere per il programma BREAKOUT apparirebbe così:

> "Inizializza le variabili; stampa il muro nella parte alta dello schermo; muovi la pallina; controlla se la pallina ha colpito un muro e se è così modifica la sua direzione; controlla se la pallina ha colpito un mattone e se è così cancella il mattone e incrementa il punteggio; controlla se la pallina si trova in fondo allo schermo e se è così confronta la sua posizione con la posizione della racchetta; se le due posizioni coincidono fai rimbalzare la pallina con una angolazione casuale, altrimenti decrementa il numero di palline disponibili e controlla se sono esaurite; se è così termina il gioco, altrimenti torna alla parte che muove la pallina e prosegui".

Volendo potreste scrivere un programma come BREAKOUT basandovi solo su questa struttura. Se iniziate a scrivere il programma senza qualcosa del genere potrebbe capitarvi di terminare e di accorgervi che avete dimenticato qualcosa di importante solo quando fate girare il programma. Dovreste allora introdurre la parte mancante comprimendola tra due linee, magari aggiungendo un GO TO per ottenere la sequenza corretta e provare diverse altre volte per raggiungere il risultato finale.

Ogni volta che lo Spectrum incontra un'istruzione GO TO o GO SUB, deve scorrere l'intero programma dall'inizio, linea per linea, fin quando incontra quella richiesta dall'istruzione. Quindi, più avanti nel programma appare la linea, più lento a girare è il programma. La velocità è influenzata da questo fenomeno in misura molto piccola, ma in un programma di grafica con figure in movimento, ogni piccolo ritardo può rendere il programma realmente lento. Quindi è conveniente sistemare le routine che vengono eseguite più spesso, il più possibile vicino all'inizio del programma. Quando la velocità di un programma è importante, potete iniziare il programma con una frase REM che contenga il titolo del programma e inserire come seconda linea un'istruzione GO SUB, diciamo per esempio alla linea 9000, dove vengano inizializzate le variabili e vengano fornite all'utente tutte le spiegazioni necessarie. Questo metodo comporta due vantaggi. Prima di tutto il computer non dovrà passare in rassegna anche le linee che inizializzano le variabili e stampano le istruzioni, ogni volta

che ricerca la linea indirizzata da un GO TO o da un GO SUB. Secondariamente, se mentre scrivete il programma vi accorgete di avere dimenticato una variabile di cui avete bisogno, potete inserirla prima del RETURN della routine di inizializzazione, invece di introdurla nel mezzo del programma dove magari non avete spazio.

Un altro obiettivo da tener presente quando scrivete un programma è che tutti coloro che lo useranno devono sapere esattamente cosa fare per usarlo. Il programma dovrebbe contenere in proposito delle spiegazioni esaurienti o comunque le relative istruzioni devono essere fornite in qualche altro modo. Una serie di istruzioni scritte su carta e accompagnate da esempi (descrivendo all'utente i tasti da premere per ottenere determinati risultati) sono probabilmente migliori di una lunga serie di istruzioni inserite nel programma stesso. Se tutto questo non è molto importante per un programma scritto per uso personale, diventa vitale per un programma che deve essere venduto, o che comunque deve essere usato anche da altre persone.

È molto importante aggiungere nei vostri programmi anche dei controlli sui dati inseriti dall'utente, in modo che un input errato non provochi un disastro. I sistemi per ottenere questo sono descritti nel capitolo sull'educazione e sono menzionati anche in altre parti del libro. Tecniche che consentono all'utente di ricontrollare i dati inseriti prima di farli accettare al programma (come accade in alcuni programmi del capitolo sugli affari) sono importanti quando viene richiesta una grande quantità di dati da tastiera. È una buona idea effettuare un controllo finale e consentire la correzione dei dati prima che vengano definitivamente accettati. Anche questa caratteristica è presente nei programmi descritti nel capitolo sugli affari.

Inoltre i messaggi che appaiono sul video dovrebbero essere più chiari possibile. Un cursore lampeggiante in fondo allo schermo indica che il computer è in attesa di un dato, ma fino a quando non viene stampato qualcosa sullo schermo o non viene aggiunto un messaggio all'istruzione di input, l'utente potrebbe non sapere quale dato gli viene richiesto.

Una lista a parte dei dati da introdurre non sostituisce in nessun modo un uso appropriato dei messaggi del programma. Molti programmi scritti sul predecessore dello Spectrum, lo ZX81, non potevano permettersi il lusso di messaggi appropriati, a causa della mancanza di memoria, ma questa scusa, in genere, non è valida sullo Spectrum.

Le frasi REM dovrebbero spiegare la funzione della sezione di programma che segue, in modo da avere un listato chiaro; assicurandovi che tornando ad esaminare il programma dopo un certo periodo di tempo siate subito in grado di conoscere cosa dovrebbe fare una determinata parte del programma. Nel capitolo sull'educazione, il programma GATTI E COSE vi mostra un uso corretto della frase REM. Anche solo guardando le frasi REM, dovreste essere in grado di dire cosa fa un certo programma; se è necessario dovreste poter ricostruire il programma partendo dalle frasi REM: SCEGLI E FORMA GLI OGGETTI; STAMPA IL NUMERO DEGLI OGGETTI; CORREZIONE; ELOGIO; INPUT DI UN TASTO; DATI. Non c'è dubbio che un programma che contenga frasi REM è più semplice da esaminare di uno che non ne contiene affatto.

Una volta che avete stabilito cosa deve fare il programma e avete compilato una lista delle sue parti principali, fate un elenco delle parti che compongono ogni sezione. Suddividendo ogni routine in tante routine più piccole, scoprirete che il programma si scrive praticamente da solo. Vi accorgerete anche di possibili errori od omissioni, prima ancora di avere introdotto una singola linea di programma. Potrebbe valere la pena di tenere un elenco delle variabili usate su un foglio di carta, intanto che preparate il programma. Così, quando arriverete alla sezione del programma che inizializza le variabili, ne avrete una lista già pronta.

Troverete più facile capire cosa sta succedendo in un programma se userete nomi di variabile espliciti, come PUNTI per tenere il punteggio e MAXPUNTI per il punteggio record. Sebbene questo richieda un po' più di tempo che usare nomi con una sola lettera, minimizza la possibilità di usare lo stesso nome per due cose diverse, in un programma lungo; questo errore darebbe molte difficoltà in fase di correzione.

Inoltre i nomi di variabili lunghi richiedono più memoria dei nomi formati da una sola lettera, ma questo non è un problema per la maggior parte dei programmi che scriverete sullo Spectrum.

In generale è meglio non reinventare la ruota. Sebbene non dovreste rubare o adattare programmi e poi farli passare per vostri, è inutile perdere delle ore per creare routine, tipo quelle di ordinamento, che sono già disponibili in gran numero sulle riviste. È abbastanza probabile infatti che finiate con il produrre una routine identica a quella che potevate prendere da qualche parte, dato che c'è un numero limitato di modi per svolgere certi compiti. Questo suggerimento è valido però solo se desiderate finire un programma a tutti i costi. La soddisfazione che otterrete dallo scrivere una routine partendo da zero, non importa quanto standard possa essere, e le conoscenze che ne ricaverete, vi ripagheranno più che abbondantemente del tempo impiegato. Così se il tempo non vi manca, potrebbe anche valere la pena di reinventare la ruota.

Molti programmatori diventano in poco tempo esperti di una parte del BASIC dello Spectrum, ma una volta che hanno imparato questo, mettono via il manuale e non proseguono la esplorazione degli altri comandi disponibili. Non importa quanto bene conoscerete il vostro Spectrum e la sua versione del BASIC, dovreste sempre leggere il manuale (e questo e altri libri sul computer) di tanto in tanto, proprio per vedere se c'era qualche punto che avevate male interpretato o che non conoscevate.

Potreste trovare più semplice e più istruttivo far girare mentalmente delle parti di un programma, come se foste il computer piuttosto che premere semplicemente RUN. Partite dall'inizio del programma e seguite pari pari le istruzioni che incontrate. Quando lavorate con il computer in questo modo, spesso scoprite routine scritte malamente o che potrebbero dare dei problemi e che l'esecuzione sullo Spectrum non aveva messo subito in risalto. Far girare a mano un programma è anche un buon metodo per scoprire se il computer a un certo punto si troverà a fare qualcosa come una divisione per zero, che arresterebbe il programma con un messaggio di errore.

Assicuratevi che i risultati e i messaggi del programma siano chiari per l'utente.

Non potete scrivere un programma assumendo che ogni volta che il programma verrà eseguito vi troverete a fianco dell'utente. Un programma che vi costringe a dire cose del tipo "Il numero nell'angolo in alto a destra dello schermo è il numero di tentativi fatti per indovinare la risposta" o "Il primo numero che vedi è il totale annuale delle vendite e il secondo è la proiezione per il secondo mese", potrebbe difficilmente essere descritto come ben progettato.

I suggerimenti che trovate in questo capitolo sono di due tipi:

- Cose da tener presenti prima di scrivere il programma.
- Interazione con l'utente che non conosce il programma (sia per i dati da inserire che per i risultati da stampare).

Se scriverete i vostri programmi con l'intenzione di sfruttare al meglio queste due aree scoprirete probabilmente che la qualità del vostro lavoro migliorerà immediatamente e continuerà a migliorare.

SUGGERIMENTI PER ULTERIORI LETTURE:

- *"The Programmer's Book of Rules"* — Ledin George e Victor uLifetime Learning Pubblications, USA 1979)
- *"BASIC Programmer's Notebook Savage"* — Earl R. (Howard Sams & Co. Inc, USA 1982)
- *"BASIC With Style"* — Nagin Paul A. e Ledgard Henry F. (Hayden Book Company Inc, USA 1978)
- *"Guide To Good Programming Practice"* — Meek Brian e Heath Patricia (eds) (Ellis Horwood Ltd./John Wiley & Sons, 1980)

APPENDICE

STORIA

Il vostro Spectrum rappresenta una delle ultime tappe della lunga strada che l'uomo ha percorso per costruire i calcolatori. La prima 'macchina' fu probabilmente l'abaco, che sfruttava delle aste parallele fissate su un piano. La posizione delle palline sulle aste indicava un numero particolare. L'abaco si usava ancora fino a poco tempo fa, quando è stato sostituito dalla calcolatrice tascabile.

John Napier, nato in Scozia, fu una delle prime figure che possiamo identificare con la costruzione di macchine in grado di eseguire calcoli. Napier inventò uno strumento divenuto famoso con il nome di "Napier's Bones" (letteralmente macinatrice di calcoli), nove bastoncini rettangolari che consentivano di eseguire delle moltiplicazioni sommando assieme i numeri sui bastoncini. Il pastore protestante inglese William Oughtred, elaborò un primitivo modello di regolo nel 1621, che eseguiva moltiplicazioni sommando le lunghezze sul regolo relative ai logaritmi dei numeri, lo stesso principio usato da tutti i regoli costruiti in seguito.

Più tardi nel XVII secolo, Blaise Pascal sviluppò un metodo per sommare i numeri che faceva uso di ingranaggi collegati. Egli predispose gli ingranaggi in una scatola, dotata di piccole finestre per consentire di vedere il risultato di un calcolo, dopo aver 'composto' i numeri da sommare.

Gottfried Leibniz, un contemporaneo di Pascal, inventò un sistema per eseguire moltiplicazioni automaticamente. Il suo metodo era così buono che veniva ancora usato 250 anni più tardi nelle macchine calcolatrici.

Nel 1792 nasce nel Devon Charles Babbage. Egli fu una delle più importanti figure nella storia dei computer. In effetti molti sostengono che egli abbia inventato il primo vero calcolatore. Babbage si trovava male con le tavole matematiche disponibili a quei tempi, così provò a costruire una macchina differenziale per elaborare calcoli accurati. Tuttavia fu ostacolato dall'inesperienza dei tecnici suoi contemporanei che non erano in grado di costruire le parti con la precisione richiesta. Babbage perse interesse per la macchina differenziale e iniziò la costruzione di una macchina 'analitica' che, se avesse funzionato come nei suoi progetti, avrebbe rappresentato il primo computer della storia; una macchina calcolatrice in grado di portare a termine qualunque tipo di calcolo che avrebbe anche potuto prendere delle decisioni sulle operazioni da svolgere in base ai risultati sui calcoli.

Dietro alla capacità dei computers di prendere decisioni sta un ramo ben preciso della matematica, detto algebra di Boole. Fu concepita dal matematico e logico in-

glese George Boole molto prima che si pensasse di applicarla ai moderni computer. Nel 1890 furono usate per la prima volta delle schede perforate per registrare e catalogare i dati del censimento americano. Il sistema a schede fu progettato da Herman Hollerith, che fondò una ditta per vendere il suo sistema. La compagnia si ingrandì, assorbì molti dei suoi concorrenti e infine fu ribattezzata International Business Machine Corporation ed oggi è la più grande industria al mondo di calcolatori, la IBM.

Intorno al 1870, un fisico inglese, Lord Kelvin, inventò una macchina per predire il flusso delle mareee e in seguito studiò un dispositivo chiamato 'analizzatore differenziale' che una volta costruito avrebbe permesso di risolvere non solo i problemi connessi alle maree, ma qualunque problema legato alla soluzione di equazioni differenziali. Una macchina simile fu costruita nel 1930 da un professore del MIT, Massachusetts Insitute of Technology, Vannevar Bush. Sebbene la macchina funzionasse, Bush capì che non ci poteva essere un futuro per le macchine calcolatrici basate su dispositivi meccanici; così sostituì alcune parti con dei circuiti a valvole. Era stato compiuto un altro passo verso i moderni computer.

Negli anni 40, George Stibitz, che lavorara per la Bell Telephone Laboratories, scopre che le informazioni binarie (informazioni trattate come una serie di 0 e 1), potevano essere rappresentate e manipolate con degli interruttori; un interruttore chiuso rappresentava l'uno e uno aperto rappresentava lo zero. Un dispositivo che egli realizzò con relays telefonici e piccole lampadine fu il primo calcolatore elettronico del mondo e aprì la strada per usare l'aritmetica binaria nei computer. Quasi tutti i computer moderni fanno uso dell'aritmetica binaria che viene tradotta dalla macchina in numeri e lettere a noi comprensibili.

Sebbene l'idea del primo computer fu sviluppata a Cambridge, in Inghilterra, nel 1832 fu solo nel 1944 che essa venne effettivamente realizzata, a Cambridge, Massachusetts, in America. La prima macchina calcolatrice completamente automatica, 'Automatic Sequence Controlled Calculator', fu completata in quel anno da Hozard Aiken di Havard, che lavorava per l'International Business Machine Corporation.

I primi computer erano molto grandi, consumavano grandi quantitativi di elettricità per alimentare le valvole ed erano notoriamente poco affidabili. Il transistor, inventato nel 1947, ridusse di molto le dimensioni dei computer, ma fu solo con la realizzazione del primo microprocessore (ciò che noi ora chiamiamo 'chip') nel 1971, che i minuscoli computer come lo Spectrum divennero possibili.

L'idea del circuito integrato, il precursore del chip, fu suggerita per la prima volta da G.W. Dummer, che lavorava per il Royal Radar Establishment. Nessuno fece molto caso alla notizia finchè sei anni più tardi in America, Jack Kilby, della Texas Instruments riuscì per la prima volta a costruirne uno.

La compagnia americana Intel realizzò il primo microprocessore della storia, noto come 4004, nel novembre del 1971. La Intel e il mondo non si resero subito conto dell'enorme impatto sociale che ne sarebbe derivato. Ci volle un anno perchè la

gente cominciasse a rendersi conto che le basi di una rivoluzione che avrebbe cambiato il mondo erano state gettate.

Il primo personal computer, l'Altair, fu costruito da una piccola ditta del Nuovo Messico, la Mits, nel 1975. Era cominciata la gara per produrre personal computer sempre più piccoli e sempre più potenti. Clive Sinclair si lanciò in quella corsa negli anni '70.

Egli aveva fondato la sua compagnia, la Sinclair Radionics, nel 1962 per produrre radio e amplificatori in scatole di montaggio da vendere per posta dietro inserzioni pubblicitarie. Per il 1967 la compagnia aveva raggiunto un giro d'affari di 100.000 sterline all'anno e la gamma dei suoi prodotti era arrivata a includere l'alta fedeltà.

Cinque anni più tardi Sinclair entrò nel mercato delle calcolatrici con la Executive, la prima calcolatrice tascabile del mondo che lasciò tutti sbalorditi. Nel gennaio del XOU7 i mezzi di comunicazione rendevano noto il lancio del Sinclair's Microvision, la prima televisione tascabile, con un minuscolo schermo da due pollici. Due anni più tardi veniva venduta una versione per il mercato inglese a meno della metà del prezzo originale.

Alla fine del gennaio del 1980, Clive Sinclair realizzò il suo primo computer, lo ZX80. Era a quell'epoca il computer più economico del mondo e sembra che abbia giocato un ruolo fondamentale nella diminuzione dei prezzi dei piccoli computer fino ad oggi. Con il lancio dello ZX80, ancora piuttosto limitato, nel 1980 Sinclair divenne in pochissimo tempo il più grande produttore di microcomputer della Gran Bretagna. Con la realizzazione del suo successore, il più flessibile ZX81, soltanto un anno dopo, Sinclair raggiunse il titolo mondiale.

Lo Spectrum, che è stato lanciato nel Regno Unito nell'aprile del 1982 è un degno successore dello ZX81; offre infatti la stessa potenza di calcolo (sebbene giri molto più velocemente) con in più caratteristiche quali il colore, il suono e l'alta risoluzione grafica. La sua realizzazione ha portato a una nuova riduzione dei prezzi su vasta scala e ben presto sono stati annunciati diversi rivali dello Spectrum.

SUGGERIMENTI PER ULTERIORI LETTURE:

- *"Bit – PS – NG "* – Sperimentore e altri libri Spectrum Saccoov
- *"The Making of the Micro"* – Evans, Christopher (Gollancz, 1982)
- *"Early British Computers"* – Lavington, Simon (Manchester University Press, 1980)
- *"The Personal Computer Guide"* – Hartnell Tim (Virgin Books, 1982)
- *"Personal Computer Handbook"* – Buchsbaum, Walter H. (Hovard W Sams & Co., Inc. USA, 1981)
- *"The Personal Computer Book"* – Bradbeer, Robin (Gower Publishing Co., 1982)

— *"A Simple Guide to Home Computers"* Ditlea, Steve (A & W Visual Library, USA 1979)

— *"Getting Involved With Your Own Computer"* — Solomon, Leslie and Veit, Stanley (Enslow Publishers, USA, 1977)

— *"Personal Computing, a Beginner's Guide"* — Bunnell David (Dutton, USA, 1978)

PERIFERICHE

Periferica è un vocabolo del linguaggio tecnico che letteralmente significa "qualcosa collegato al computer che non fa realmente parte di esso". Le periferiche più importanti sono i dispositivi di ingresso a disposizione dell'uomo (per esempio la tastiera, incorporata in molti computer dei nostri giorni), i dispositivi di uscita (video e stampanti) e le memorie di massa (come dischi o nastri). Molti computer di grandi dimensioni dispongono di altri computer più piccoli che regolano il collegamento con l'esterno (peripheral processor, controllori di periferiche). Questi computer satelliti sono in genere molto più grandi dello Spectrum.

È disponibile un'interfaccia seriale RS232 per lo Spectrum, che vi consente di collegare il vostro calcolatore con una vasta gamma di stampanti, terminali e altri computer. La RS232 è una delle interfacce standard disponibili più diffuse, ecco perchè sono stati prodotti così tanti dispositivi compatibili RS232.

STAMPANTI

Una stampante è necessaria per quasi tutte le applicazioni professionali. Ecco un elenco di vari tipi di stampanti e dei termini tecnici più comuni in questo settore:

Line Printer — Stampante seriale: un tipo di stampante grande e costosa che stampa una linea alla volta, di solito con una velocità massima di 600 linee (132 caratteri) al minuto. Viene impiegata con computer di grandi dimensioni.

Daisywheel e golfball — Stampante a Margherita: i caratteri vengono stampati con una testina simile a quella delle macchine da scrivere.

Dot matrix — Matrice di punti: i caratteri sono formati da una matrice di aghi e vengono stampati termicamente o con un procedimento elettrostatico.

Impact — Impatto: il meccanismo di stampa colpisce un nastro inchiostrato che segna la carta.

Thermal — Termiche: lavorano su una carta speciale che annerisce col calore; la stampante riscalda molto rapidamente un singolo puntino e subito dopo lo raffredda.

Electrostatic – Elettrostatiche: lavorano con carta d'argento ricoperta da un sottile strato di alluminio; la stampante genera delle scintille che impressionano la carta lasciando una traccia visibile.

Alta qualità e velocità della stampa sono cose che si pagano. La carta termica e quella elettrostatica sono molto care, sebbene le stampanti di questo tipo siano abbastanza economiche. Se desiderate semplicemente ottenere in stampa piccole quantità di dati al costo più basso possibile, la stampante Sinclair è molto valida. Quando questo libro è stato scritto il costo delle stampanti elettrostatiche in Gran Bretagna partiva da 60 Sterline (Stampante Sinclair, 32 caratteri per linea), quello delle stampanti a impatto dotmatrix di tipo standard da 200 Sterline, le stampanti termiche erano leggermente più economiche. Le stampanti a margherita partivano da 300 Sterline e arrivano attorno alle 1500 per i modelli più veloci.

MEMORIE DI MASSA (MSD – Mass Storage Devices)

Le memorie di massa più comuni sono i nastri e i dischi magnetici. Per registrare i programmi dello Spectrum su cassetta conviene usare registratori portatili mono. Cercate di evitare registratori con testine stereo. Se possibile procuratevene uno che sia noto per dare buoni risultati con i segnali provenienti dal computer (non deve necessariamente trattarsi di un dispositivo costoso). Usate l'alimentazione a rete piuttosto che quella a batteria (se ottenete risultati scarsi con l'alimentazione a rete cambiate registratore). Per questa applicazione si rivelano utili alcuni degli accessori disponibili su certi registratori: il contagiri; controllo manuale di volume (al posto di quello automatico che spesso è impreciso); strumento indicatore del livello di registrazione (level meter); possibilità di far avanzare il nastro e di farlo tornare indietro; un idoneo monitor per controllare la registrazione e tenere sotto controllo gli eventuali sbalzi di volume o distorsioni. Un controllo del tono si può definire indesiderabile; se il vostro registratore ne ha uno posizionatelo sul massimo. In molti casi è preferibile non collegare contemporaneamente le prese "microphone" e "earphone".

Usate delle buone cassette, mai più lunghe di una C60. Le TDK tipo D sono eccellenti. BASF, Agfa, ecc. producono tutte nastri soddisfacenti. Cassette C12 e C15 vendute per registrazioni effettuate dal computer sono il meglio, ma non tutti i nastri venduti sono di qualità abbastanza buona. Un monitor o un level meter sul registratore vi aiuteranno a valutare le qualità del nastro verificando eventuali fluttuazioni del segnale.

I dischi sono molto più veloci dei nastri. Consentono fra l'altro di reperire informazioni ovunque siano state memorizzate sul disco in modo rapido, senza bisogno di effettuare operazioni di riavvolgimento (come accade per i nastri). Potete leggere diversi file in sequenza molto rapidmente ed è anche possibile alternare operazioni su file diversi che risiedano sullo stesso disco (esempio,leggere informazioni relative a un prodotto del magazzino, aggiornarle e registrarle in un altro file, o anche sullo stesso. Su un nastro questo sarebbe stato impossibile: avrebbe richiesto continui avvolgimenti del nastro per raggiungere determinati punti).

Il Microdrive, un floppy disk in miniatura, può contenere fino a 100 K di dati. Si possono collegare fino a 8 di questi dischetti allo Spectrum contemporaneamente. I dati sono trasferiti dal Microdrive allo Spectrum (e viceversa) all'incredibile velocità di 16 K al secondo, quando il transfer rate (velocità di trasferimento dei dati) per un registratore a cassette è di circa 1.5 K al secondo.

Le memorie di massa non sono mai sicure al 100%; è vitale tenere delle copie dei dati importanti (backup). Per certi volumi di dati, i tabulati possono essere usati come estrema difesa delle informazioni; naturalmente non potete fare affidamento su questo metodo se c'è il rischio che dobbiate reinserire manualmente gli ordini di tre mesi. I backup non devono essere realizzati sulla MSD che viene usata per lavorare; i dischi possono ad esempio essere copiati su nastri. Tenete i backup in un luogo differente dalle unità normali. Ricordate che spesso le informazioni memorizzate possono avere più valore di tutto il sistema. Se viaggiate in treno o in metropolitana non appoggiate supporti magnetici sul pavimento vicino ai motori. Non assumete mai che il disco o il nastro che state per cancellare o riscrivere sia quello giusto senza effettuare un controllo.

SPECIFICHE DELLO SPECTRUM

DIMENSIONI

Larghezza 233 mm.Lunghezza 144 mm.
Altezza 30 mm.

CPU/MEMORIE

Microprocessore Z80 operante a 3.5 MHz. 16 K di ROM contenente l'interprete BASIC e il sistema operativo. 16 K di RAM (più 32 K di espansione su scheda interna) o 48 K.

TASTIERA

40 Tasti con maiuscole e minuscole con possibilità di CAPS LOCK. Tutte le parole BASIC ottenibili da singoli tasti, più 16 caratteri grafici, 22 codici di controllo per il colore e 21 caratteri grafici definibili dall'utente. Tutti i tasti hanno l'auto repeat.

DISPLAY (Output su video)

Display di 256 x 192 punti memory-mapped; un byte di attributi per carattere che definisce uno degli otto colori di sfondo, uno degli otto colori di testo, carattteristiche di luminosità e lampeggio. Otto colori diversi selezionabili anche per il bordo dello schermo. Segnale di uscita per televisori a colori con sistema PAL o in bianco e nero (in questo caso i colori appaiono come diverse tonalità di grigio) sulla banda UHF canale 36.

SUONO

Altoparlante interno controllabile con il comando BASIC BEEP, 10 ottave (130 semitoni). Prese Jack sul retro del computer per collegare un amplificatore esterno.

GRAFICA

Comandi per disegnare punti, linee, cerchi ed archi di cerchio in alta risoluzione. 16 caratteri grafici pre-definiti più 21 definibili dall'utente. Funzioni per stampare in una posizione voluta dello schermo, per conoscere gli attributi di una data posizione sullo schermo (colori, luminosità e lampeggio) e per sapere se un singolo punto è acceso o no. Il testo compare sul video in 24 linee di 32 caratteri. Testo e grafica possono essere liberamente mischiati.

COLORI

I colori del testo e dello sfondo, la luminosità e il lampeggio sono controllati dai comandi BASIC INK, PAPER, BRIGHT e FLASH. Il comando OVER può essere usato per sovrapporre (con un OR Esclusivo) caratteri o figure sullo schermo. La funzione INVERSE produce una stampa con i colori di testo e sfondo scambiati.

Questi sei comandi possono essere usati in modo generale per influenzare le successive operazioni di PRINT, PLOT, DRAW e CIRCLE, o in modo specifico per agire solo sul risultato di una di queste istruzioni. Possono anche essere usati per influenzare il testo stampato da una fase INPUT. I codici di controllo per il colore sono accessibili da tastiera e possono essere inseriti nel testo o nel listato dei programmi; quando vengono eseguiti influenzano tutte le successive operazioni fino a quando non interviene un nuovo controllo sul colore. Analogamente per la luminosità e il lampeggio. I codici di controllo del colore non influenzano i listati dei programmi. Il colore del bordo viene controllato dal comando BORDER. Gli otto colori disponibili sono nero, blu, rosso, magenta, verde, azzurro, giallo e bianco. Tutti gli otto colori possono apparire sullo schermo contemporaneamente, con alcune aree lampeggianti e altre fisse e ogni area può essere resa luminosa.

VIDEO

Il video è diviso in due sezioni. La parte alta, di solito le prime 22 linee, serve per i listati, i risultati dei programmi e l'esecuzione dei comandi. La parte bassa, di solito le ultime 2 linee, viene utilizzata per le linee che si stanno introducendo o per le linee che vengono corrette. Viene usata anche per i messaggi del sistema. In fase di correzione si possono usare comandi per spostare il cursore, per inserire o cancellare caratteri (con possibilità di auto repeat). La parte bassa del video si espande fino a 22 righe se la linea da inserire richiede molto spazio.

OPERAZIONI MATEMATICHE E FUNZIONI

Sono disponibili le normali operazioni aritmetiche +, −, * e / più l'elevamento a potenza. Le funzioni matematiche seno, coseno, tangente e le loro inverse; logaritmi

neperiani ed esponenziali; funzione signum, valore assoluto e parte intera; radice quadrata, generatore di numeri casuali e valore di pi greco.

I numeri sono memorizzati in 5 byte in notazione floating proint, con un range di $-3 \times 10^{-39} / +3 \times 10^{-39}$ e $-7 \times 10^{38} / +7 \times 10^{38}$ con una precisione di 9.5 cifre decimali. I numeri binari possono essere introdotti direttamente con la funzione BIN. I simboli =, >, <, >=, <= e <> possono essere usati per confrontare stringhe, valori aritmetici o variabili del tipo vero-falso. Gli operatori logici AND, OR e NOT producono risultati booleani ma accettano 0 (falso) e qualunque altro numero (vero).

Le funzioni definibili dall'utente si possono costruire con l'istruzione DEF FN e richiamare usando FN. Possono essere usati fino a 26 argomenti numerici o 26 stringhe e la funzione può produrre un risultato numerico o una stringa.

Per controllare i dati si possono usare i comandi READ, DATA e RESTORE.

È possibile programmare un orologio in tempo reale.

OPERAZIONI DI STRINGA E FUNZIONI

Le stringhe possono essere concatenate con l'operatore +. Le variabili di stringa possono essere confrontate per mezzo dei simboli =, >, <, >=, <=, <> e produrre risultati booleani. Le funzioni di stringhe sono VAL, VAL$, STR$, LEN. CHR$ e CODE convertono i numeri in caratteri e viceversa usando il codice ASCII. È possibile sfruttare meccanismi di composizione delle stringhe nella forma a$ (x TO y).

NOMI DI VARIABILE

Numeriche - Qualunque stringa che inizia con una lettera (non c'è distinzione tra lettere minuscole e maiuscole e gli spazi sono ignorati).

Di Stringa – Da A$ a Z$.
Cicli FOR/NEXT – Da A a Z.
Matrici di stringa – Da A$ a Z$.

Variabili semplici e matrici con lo stesso nome sono consentite e distinte le une dalle altre.

MATRICI

Le matrici possono essere multidimensionali; gli indici partono da 1. Le matrici di stringa possono non avere l'ultimo indice, cioè il numero di caratteri.

VALUTAZIONE DI ESPRESSIONI

Un'espressione viene sempre valutata durante l'esecuzione di un programma.

Questo consente l'uso di espressioni per gli argomenti delle frasi come GO TO e GO SUB. Opera anche su comandi diretti permettendo che lo Spectrum lavori come una calcolatrice.

INTERFACCIA PER IL REGISTRATORE

Un suono di controllo viene registrato prima dell'informazione per evitare possibili fluttuazioni del livello di registrazione; per rimuovere il rumore di fondo viene impiegato un trigger di Schmitt. Tutti i dati registrati iniziano con una etichetta che contiene informazioni sul tipo sulla lunghezza e sull'indirizzo dei dati oltre al titolo. Programmi, contenuti dello schermo, blocchi di memoria, stringhe e matrici di caratteri possono essere registrati separatamente. Programmi, blocchi di memoria e matrici possono essere verificati dopo la registrazione. Programmi e matrici possono essere fusi, una volta prelevati dal nastro, con il contenuto della memoria. Quando due numeri di linea o due nomi di variabili coincidono il vecchio elemento è sostituito dal nuovo. I programmi possono essere salvati con un numero di linea da cui deve riprendere l'esecuzione dopo il caricamento. La interfaccia lavora alla velocità di 1500 baud attraverso due jack da 3.5 mm.

PORTA PER L'ESPANSIONE

Questa porta è usata per interfacciare la ZX Printer, la interfaccia RS 232 e gli ZX Microdrives usando i bus dei dati, degli indirizzi e di controllo dello Z80A. I comandi IN e OUT sono l'equivalente per le porte I/0 di PEEK e POKE.

COMPATIBILITA' CON LO ZX81

Il BASIC dello ZX 81 è in pratica una versione ridotta di quello dello Spectrum. Le differenze sono le seguenti:

- *FAST e SLOW:* lo Spectrum opera alla velocità dello ZX81 in modo FAST con i display in modo SLOW e quindi non dispone di questi comandi.
- *SCROLL:* lo Spectrum esegue lo scrolling del video automaticamente, chiedendo all'operatore "scroll?" ogni volta che viene riempito lo schermo.
- *UNPLOT:* lo Spectrum può simulare la funzione UNPLOT con il comdando PLOT OVER.
- *Caratteri:* lo Spectrum usa il codice ASCII mentre lo ZX81 ha un codice non standard.

"Questo è un manuale più che un libro" scrive l'autore nell'Introduzione, e vuol dire che il modo migliore per utilizzarlo è quello di leggerlo davanti al vostro Spectrum.
Gli argomenti che potrebbero essere trattati in un libro su di un computer sono un'infinità, ma affrontarli tutti non significa fare un buon libro; probabilmente significherebbe confondere le idee al lettore. Per questo le applicazioni presentate sono state scelte con attenzione, privilegiando gli aspetti più interessanti della programmazione dello Spectrum.
Il libro è per tutti; ai principianti è dedicata la prima parte, in cui si esaminano i concetti generali della programmazione. Poi si passa alle applicazioni: si aggiungono suono e colore ai programmi, si scopre lo Spectrum negli affari e nell'istruzione, si gioca e si insegna a scrivere giochi, si approfondisce la grafica fino alla produzione di figure in tre dimensioni, e si conclude con un capitolo sul linguaggio macchina, meno conosciuto del BASIC, ma senz'altro indispensabile in diverse applicazioni.

L. 18.000

Cod. 531D ISBN 88-7056-034-1